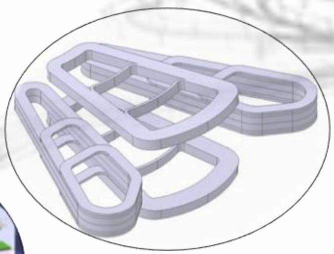
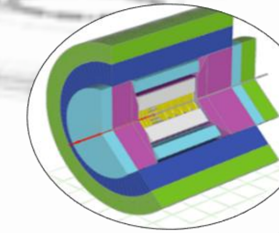
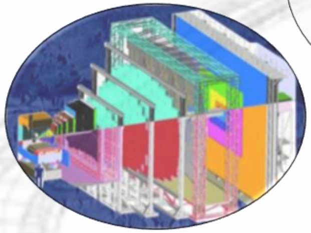
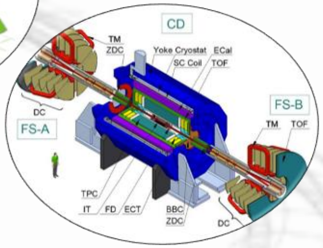
FairRoot framework

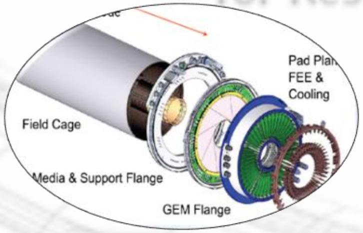


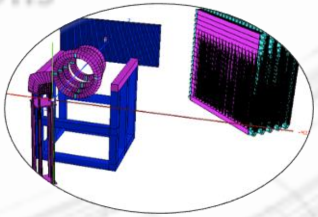






An International Accelerator Facility
for Research with Ions and Antiprotons

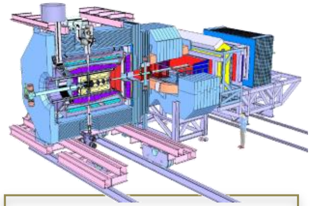




Mohammad Al-Turany
(GSI-Scientific Computing)

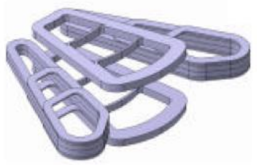
FairRoot : Timeline

Start testing the VMC concept for CBM

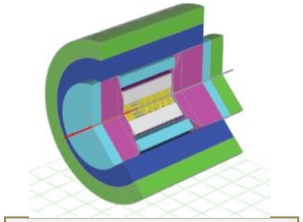


Panda decided to join->

FairRoot: same Base package for different experiments

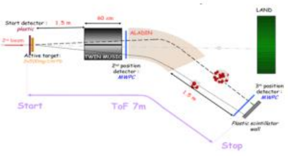


R3B joined



EIC (Electron Ion Collider BNL)

EICRoot



SOFIA (Studies On Fission with Aladin)

2004

2006

2010

2011

2012

2013

First Release of CbmRoot

MPD (NICA) start also using FairRoot

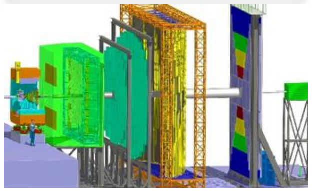
ASYEOS joined

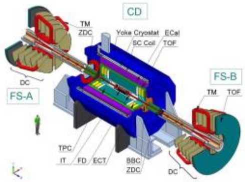
(ASYEOSRoot)

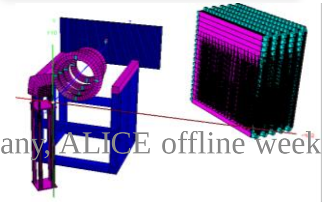
GEM-TPC seperated from PANDA branch (FOPIRoot)

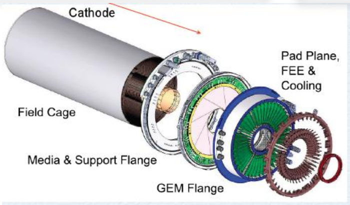
ENSAR-ROOT

Collection of modules used by structural nuclear phsyics exp.

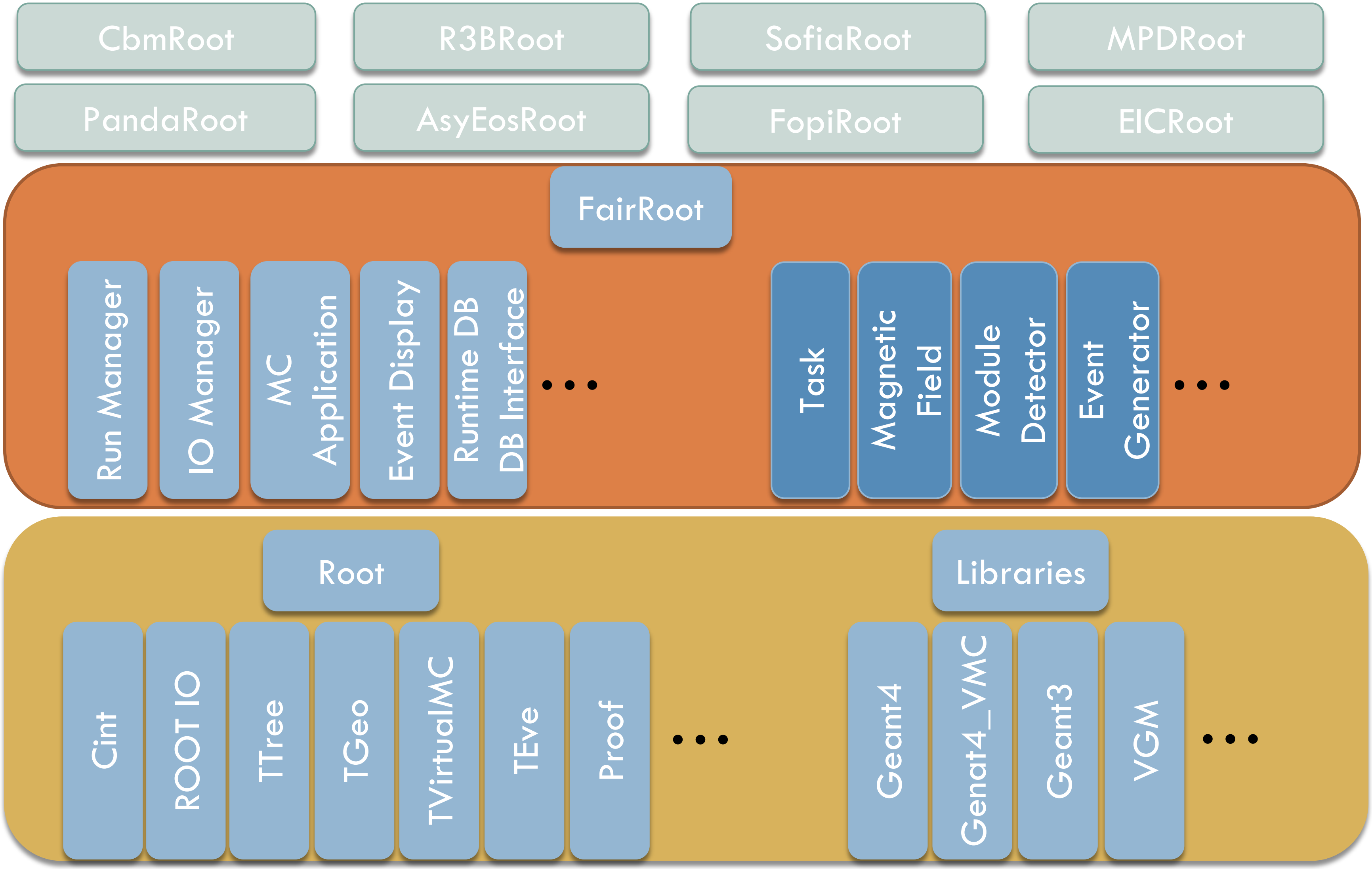








Design



CbmRoot

R3BRoot

SofiaRoot

MPDRoot

PandaRoot

AsyEosRoot

FopiRoot

EICRoot

FairRoot

Run Manager

IO Manager

MC Application

Event Display

Runtime DB
DB Interface

…

Task

Magnetic Field

Module
Detector

Event
Generator

…

Root

Cint

ROOT IO

TTree

TGeo

TVirtualMC

TEve

Proof

…

Libraries

Geant4

Genat4_VMC

Geant3

VGM

…

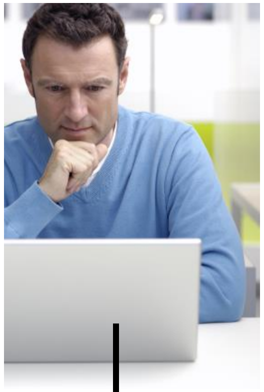
Building & Testing system

Testing and building system

- CMake
  - Creates Makefiles (and/or project files) for different platforms.
  - Test support.
  - Large user base assures support.
- CDash to handle data created with CMake
  - PHP framework
  - MySQL database
- Both tools are open source.

If someone experiments with new features in his local working copy and wants to test them (experimental build)

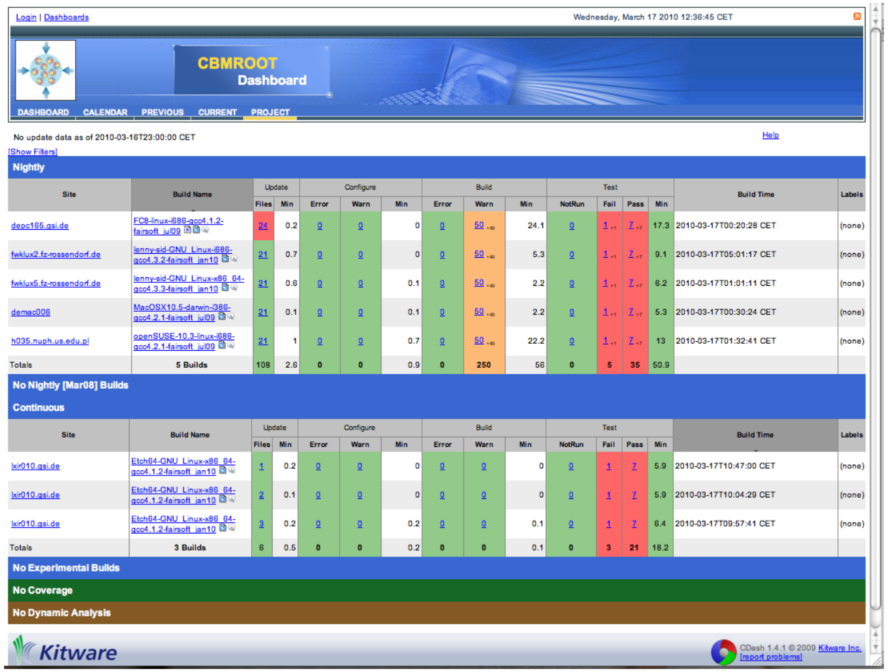
2. Configure, build and test on local machine



3. Send results automatically to central web page

1. Update (optional)

Central SVN repository



Login | Dashboards

Wednesday, March 17 2010 12:36:45 CET

CBMROOT Dashboard

DASHBOARD CALENDAR PREVIOUS CURRENT PROJECT

No update data as of 2010-03-16T23:00:00 CET

Help

[Show Filters]

Nightly

Site	Build Name	Update		Configure			Build			Test				Build Time	Labels
		Files	Min	Error	Warn	Min	Error	Warn	Min	NotRun	Fail	Pass	Min		
depc165.gsi.de	FC8-linux-i686-gcc4.1.2-fairsoft_jul09	24	0.2	0	0	0	0	50 +40	24.1	0	1 +1	7 +7	17.3	2010-03-17T00:20:28 CET	(none)
fwklux2.fz-rossendorf.de	lenny-sid-GNU_Linux-i686-gcc4.3.2-fairsoft_jan10	21	0.7	0	0	0	0	50 +40	5.3	0	1 +1	7 +7	9.1	2010-03-17T05:01:17 CET	(none)
fwklux5.fz-rossendorf.de	lenny-sid-GNU_Linux-x86_64-gcc4.3.3-fairsoft_jan10	21	0.6	0	0	0.1	0	50 +40	2.2	0	1 +1	7 +7	6.2	2010-03-17T01:01:11 CET	(none)
demac006	MacOSX10.5-darwin-i386-gcc4.2.1-fairsoft_jul09	21	0.1	0	0	0.1	0	50 +40	2.2	0	1 +1	7 +7	5.3	2010-03-17T00:30:24 CET	(none)
h035.nuph.us.edu.pl	openSUSE-10.3-linux-i686-gcc4.2.1-fairsoft_jul09	21	1	0	0	0.7	0	50 +40	22.2	0	1 +1	7 +7	13	2010-03-17T01:32:41 CET	(none)
Totals	5 Builds	108	2.6	0	0	0.9	0	250	56	0	5	35	50.9		

No Nightly [Mar08] Builds

Continuous

Site	Build Name	Update		Configure			Build			Test				Build Time	Labels
		Files	Min	Error	Warn	Min	Error	Warn	Min	NotRun	Fail	Pass	Min		
lxir010.gsi.de	Etch64-GNU_Linux-x86_64-gcc4.1.2-fairsoft_jan10	1	0.2	0	0	0	0	0	0	0	1	7	5.9	2010-03-17T10:47:00 CET	(none)
lxir010.gsi.de	Etch64-GNU_Linux-x86_64-gcc4.1.2-fairsoft_jan10	2	0.1	0	0	0	0	0	0	0	1	7	5.9	2010-03-17T10:04:29 CET	(none)
lxir010.gsi.de	Etch64-GNU_Linux-x86_64-gcc4.1.2-fairsoft_jan10	3	0.2	0	0	0.2	0	0	0.1	0	1	7	6.4	2010-03-17T09:57:41 CET	(none)
Totals	3 Builds	6	0.5	0	0	0.2	0	0	0.1	0	3	21	18.2		

No Experimental Builds

No Coverage

No Dynamic Analysis

Kitware

CDash 1.4.1 © 2009 Kitware Inc. [report problems]

4. Dashboard prepares and display results

5. Developer check results

If new code enters the central code base (continuous build)

Central SVN repository

2. Basic checks:
Style, etc

Pass

3. SVN triggers test server

Dedicated test server

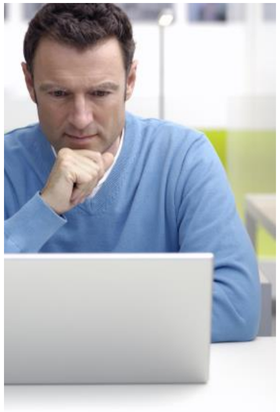
4. Configure, build and test on local machine

5. Send results automatically to central web page

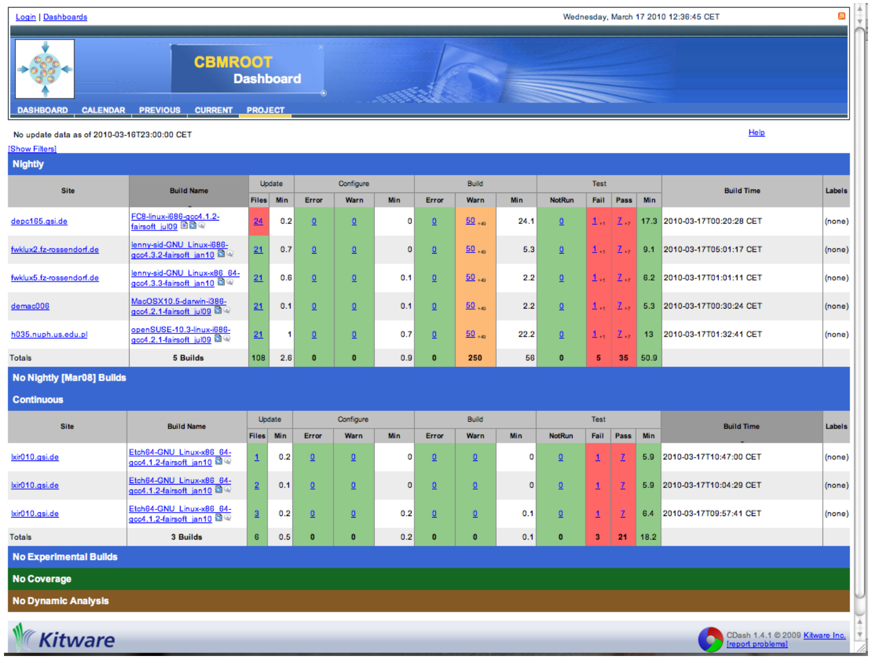
Fail

Reject commit

1. Developer commit code



7. In case of problems Dashboard sends an E-mail to Developer and Administrator



Login | Dashboards

Wednesday, March 17 2010 12:36:45 CET

CBMROOT Dashboard

DASHBOARD CALENDAR PREVIOUS CURRENT PROJECT

No update data as of 2010-03-16T23:00:00 CET

Help

[Show Filters]

Nightly

Site	Build Name	Update		Configure			Build			Test				Build Time	Labels
		Files	Min	Error	Warn	Min	Error	Warn	Min	NotRun	Fail	Pass	Min		
depc165.gsi.de	FC8-linux-i686-gcc4.1.2-fairsoft_jul09	24	0.2	0	0	0	0	50 +49	24.1	0	1 +1	7 +7	17.3	2010-03-17T00:20:28 CET	(none)
fwklux2.fz-rossendorf.de	lenny-sid-GNU_Linux-i686-gcc4.3.2-fairsoft_jan10	21	0.7	0	0	0	0	50 +49	5.3	0	1 +1	7 +7	9.1	2010-03-17T05:01:17 CET	(none)
fwklux5.fz-rossendorf.de	lenny-sid-GNU_Linux-x86_64-gcc4.3.3-fairsoft_jan10	21	0.6	0	0	0.1	0	50 +49	2.2	0	1 +1	7 +7	6.2	2010-03-17T01:01:11 CET	(none)
demac006	MacOSX10.5-darwin-i386-gcc4.2.1-fairsoft_jul09	21	0.1	0	0	0.1	0	50 +49	2.2	0	1 +1	7 +7	5.3	2010-03-17T00:30:24 CET	(none)
h035.nuph.us.edu.pl	openSUSE-10.3-linux-i686-gcc4.2.1-fairsoft_jul09	21	1	0	0	0.7	0	50 +49	22.2	0	1 +1	7 +7	13	2010-03-17T01:32:41 CET	(none)
Totals	5 Builds	108	2.6	0	0	0.9	0	250	56	0	5	35	50.9		

No Nightly [Mar08] Builds

Continuous

Site	Build Name	Update		Configure			Build			Test				Build Time	Labels
		Files	Min	Error	Warn	Min	Error	Warn	Min	NotRun	Fail	Pass	Min		
lxir010.gsi.de	Etch64-GNU_Linux-x86_64-gcc4.1.2-fairsoft_jan10	1	0.2	0	0	0	0	0	0	0	1	7	5.9	2010-03-17T10:47:00 CET	(none)
lxir010.gsi.de	Etch64-GNU_Linux-x86_64-gcc4.1.2-fairsoft_jan10	2	0.1	0	0	0	0	0	0	0	1	7	5.9	2010-03-17T10:04:29 CET	(none)
lxir010.gsi.de	Etch64-GNU_Linux-x86_64-gcc4.1.2-fairsoft_jan10	3	0.2	0	0	0.2	0	0	0.1	0	1	7	6.4	2010-03-17T09:57:41 CET	(none)
Totals	3 Builds	6	0.5	0	0	0.2	0	0	0.1	0	3	21	18.2		

No Experimental Builds

No Coverage

No Dynamic Analysis

Kitware

CDash 1.4.1 © 2009 Kitware Inc.
[report problems]

6. Dashboard prepares and display results

From time to time a full check on all supported platforms should be done (nightly build)

Central SVN repository

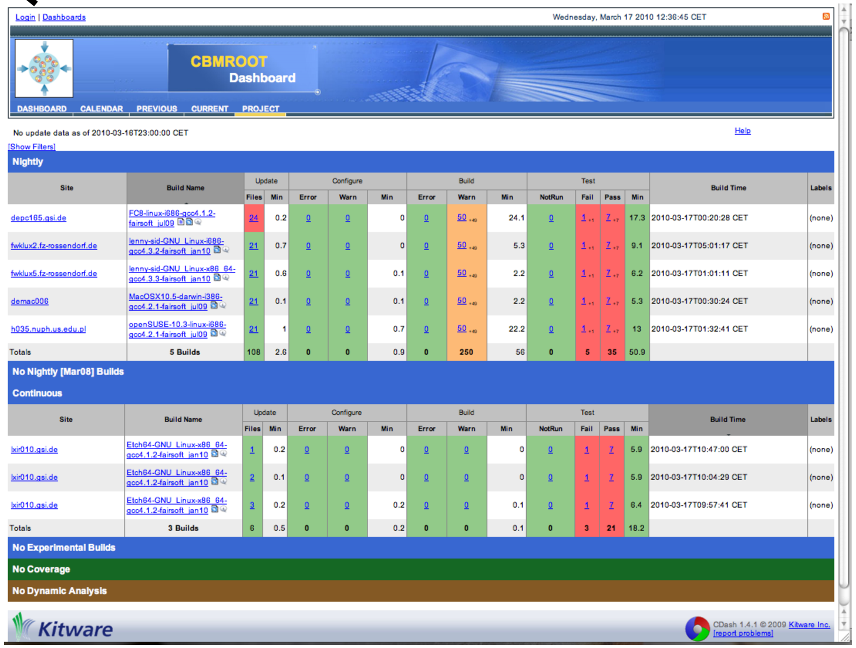
1. Update

4. In case of problems Dashboard sends an E-mail to Developer and Administrator

5. In the morning Developers and Administrators check their mails and the dashboard. And the development cycle starts again

2. Send results automatically to central web page

3. Dashboard prepares and display results



Login | Dashboards

Wednesday, March 17 2010 12:36:45 CET

CBMROOT Dashboard

DASHBOARD CALENDAR PREVIOUS CURRENT PROJECT

No update data as of 2010-03-16T23:00:00 CET

Help

[Show Filters]

Nightly

Site	Build Name	Update		Configure			Build			Test				Build Time	Labels
		Files	Min	Error	Warn	Min	Error	Warn	Min	NotRun	Fail	Pass	Min		
depc165.gsi.de	FC8-linux-i686-gcc4.1.2-fairsoft_jul09	24	0.2	0	0	0	0	50 +49	24.1	0	1 +1	7 +7	17.3	2010-03-17T00:20:28 CET	(none)
fwklux2.fz-rossendorf.de	lenny-sid-GNU_Linux-i686-gcc4.3.2-fairsoft_jan10	21	0.7	0	0	0	0	50 +49	5.3	0	1 +1	7 +7	9.1	2010-03-17T05:01:17 CET	(none)
fwklux5.fz-rossendorf.de	lenny-sid-GNU_Linux-x86_64-gcc4.3.3-fairsoft_jan10	21	0.6	0	0	0.1	0	50 +49	2.2	0	1 +1	7 +7	6.2	2010-03-17T01:01:11 CET	(none)
demac006	MacOSX10.5-darwin-i386-gcc4.2.1-fairsoft_jul09	21	0.1	0	0	0.1	0	50 +49	2.2	0	1 +1	7 +7	5.3	2010-03-17T00:30:24 CET	(none)
h035.nuph.us.edu.pl	openSUSE-10.3-linux-i686-gcc4.2.1-fairsoft_jul09	21	1	0	0	0.7	0	50 +49	22.2	0	1 +1	7 +7	13	2010-03-17T01:32:41 CET	(none)
Totals	5 Builds	108	2.6	0	0	0.9	0	250	56	0	5	35	50.9		

No Nightly [Mar08] Builds

Continuous

Site	Build Name	Update		Configure			Build			Test				Build Time	Labels
		Files	Min	Error	Warn	Min	Error	Warn	Min	NotRun	Fail	Pass	Min		
lxir010.gsi.de	Etch64-GNU_Linux-x86_64-gcc4.1.2-fairsoft_jan10	1	0.2	0	0	0	0	0	0	0	1	7	5.9	2010-03-17T10:47:00 CET	(none)
lxir010.gsi.de	Etch64-GNU_Linux-x86_64-gcc4.1.2-fairsoft_jan10	2	0.1	0	0	0	0	0	0	0	1	7	5.9	2010-03-17T10:04:29 CET	(none)
lxir010.gsi.de	Etch64-GNU_Linux-x86_64-gcc4.1.2-fairsoft_jan10	3	0.2	0	0	0.2	0	0	0.1	0	1	7	6.4	2010-03-17T09:57:41 CET	(none)
Totals	3 Builds	6	0.5	0	0	0.2	0	0	0.1	0	3	21	18.2		

No Experimental Builds

No Coverage

No Dynamic Analysis

Kitware

CDash 1.4.1 © 2009 Kitware Inc.
[report problems]

From SVN to Git

- With SVN we can only do very fast tests in the pre-commit
- Moving to Git will help us to put more tests on the code before it is committed the Master branch

Time based simulation In FairRoot

How do events overlap?

- In Detectors:
  - Sensor elements are still blocked from previous hits
  - Electronic is still busy
  - Hits too close in time cannot be distinguished
  - …
- Special problem for CBM and PANDA:
  - Continuous beam with Poisson statistics (?) ➔ many events with short time between them
  - No hardware trigger
  - Complex event reconstruction
  - ➔ Necessary to simulate data stream as realistic as possible

Time based simulation: Implementation

- FairWriteoutBuffer is Special buffer to store detector data between different events
- You give the data you want to store an absolute time window this data is active in your detector and can influence later events.
- If the same detector element is hit a second time the data is modified.
- This is an abstract base class where you have to inherit from

Time based simulation: Reading back data

- FairRootManager has new reading algorithms, which make it possible to use the event wise implemented tasks to run on such data streams
- Different algorithms available to extract data:
  - All data up to a given time
  - All data in a time window
  - All data between time gaps of a certain size

- Other algorithms can be (easily) implemented

Time-based simulation in FAIRROOT was presented in details with code

https://indico.gsi.de/contributionDisplay.py?contribId=9&confId=1810

Fast Simulation

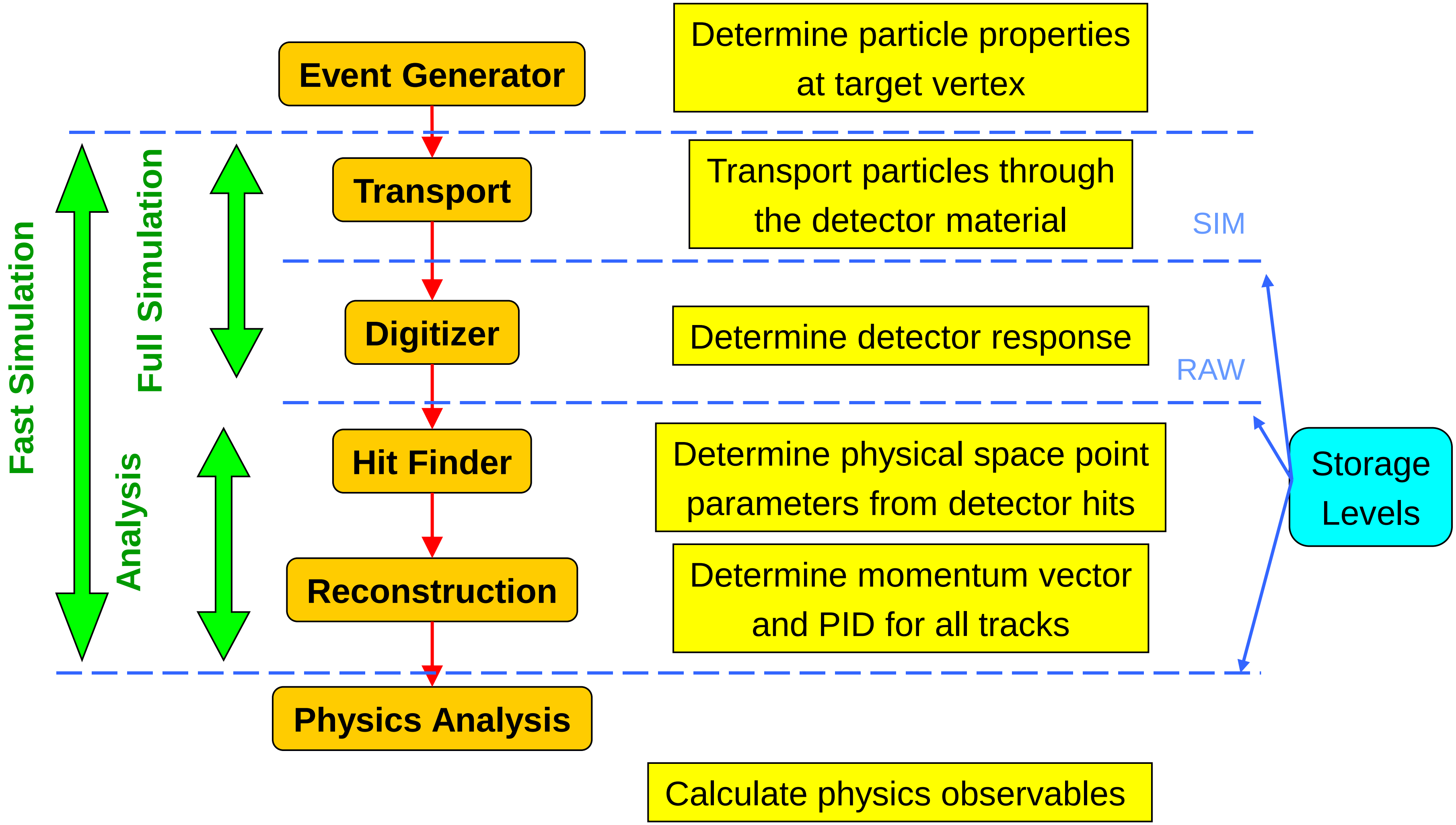
- The same application, just different configuration:

  – Event generators just push the event into the stack, no transport is taking place

  –Detector response is presented as FairTasks (TTask)

  –The output has the same form as full simulation

Simulation-Reconstruction Chain



Event Generator

Determine particle properties at target vertex

Transport

Transport particles through the detector material

SIM

Digitizer

Determine detector response

RAW

Hit Finder

Determine physical space point parameters from detector hits

Reconstruction

Determine momentum vector and PID for all tracks

Physics Analysis

Calculate physics observables

Storage Levels

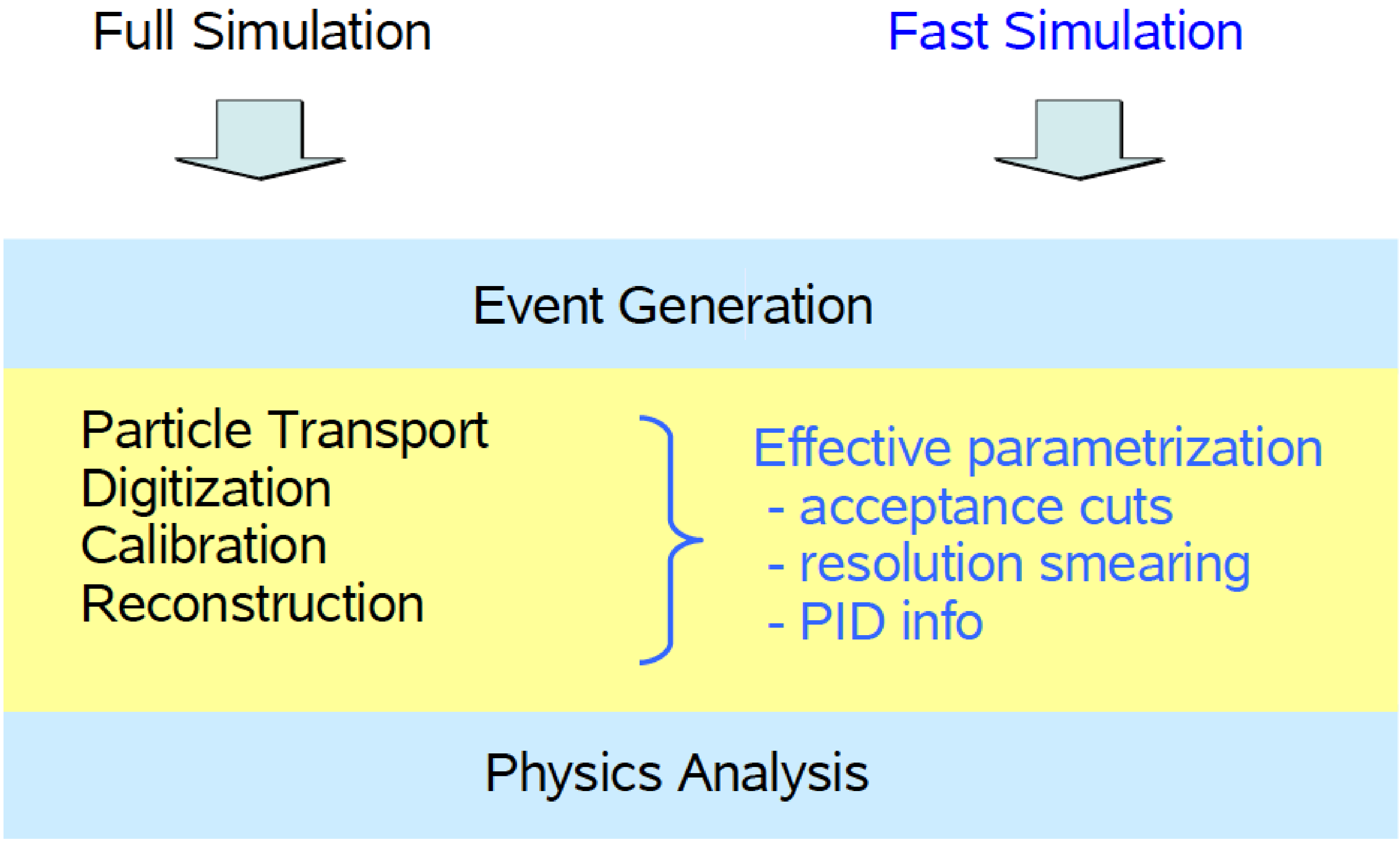
Fast Simulation

Full Simulation

Analysis

Klaus Götzen

Fast Simulation: Concept



Full Simulation

Fast Simulation

Event Generation

Particle Transport
Digitization
Calibration
Reconstruction

Effective parametrization
- acceptance cuts
- resolution smearing
- PID info

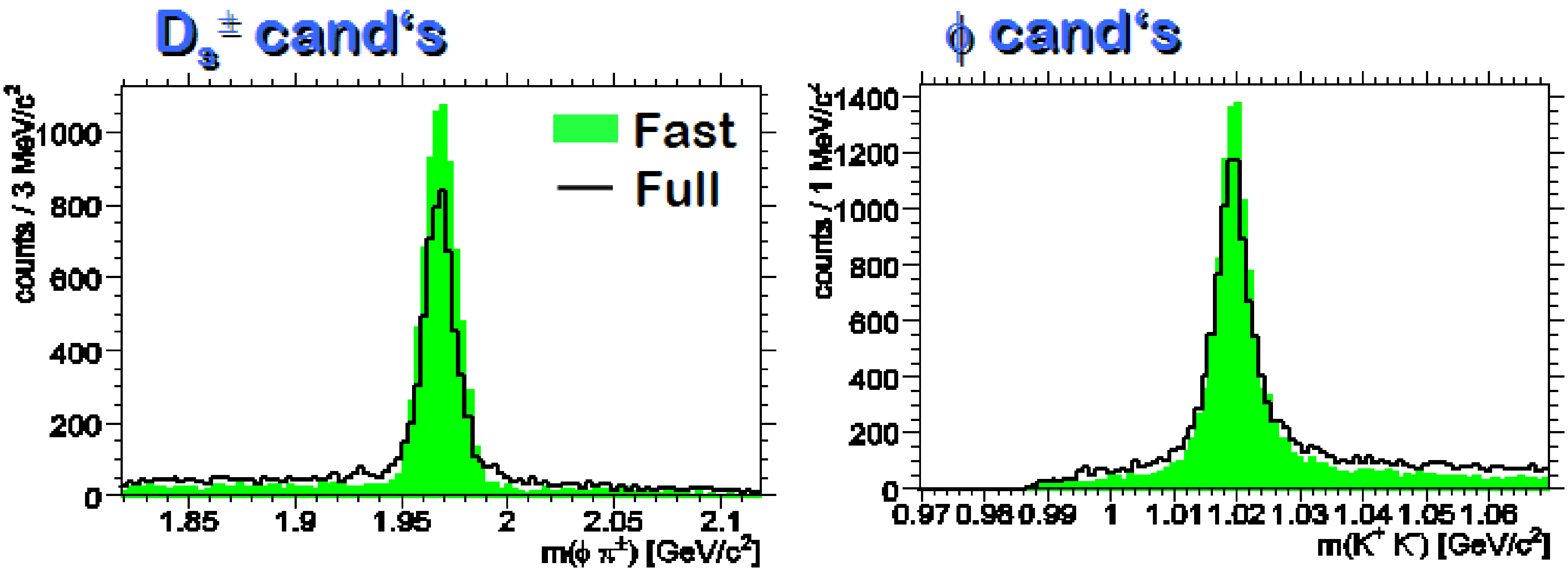
Physics Analysis

Compared to Full Sim

Klaus Götzen

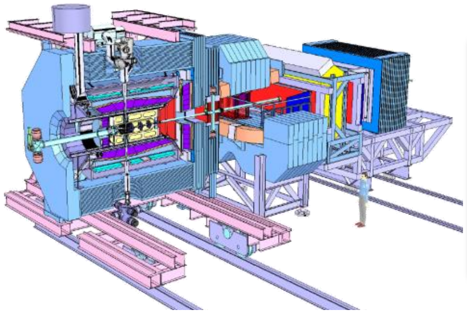
- Comparison to Full Sim are reasonable

  (channel: $\bar{p}p \rightarrow D_s D_{so}$)

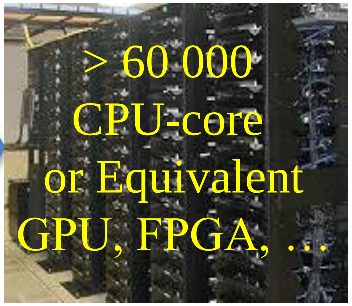


(10 k Signal events; absolute numbers)

Next challenge is: Online vs. Offline or Online + Offline ?



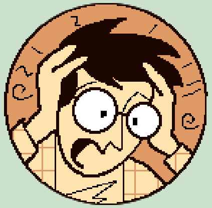
300 GB/s
20M Evt/s

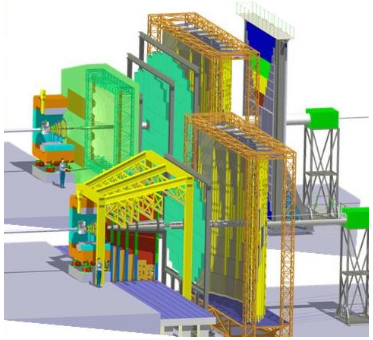


> 60 000
CPU-core
or Equivalent
GPU, FPGA, …

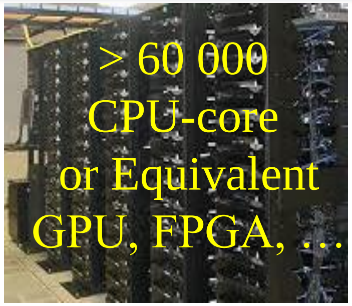
< 1 GB/s
25K Evt/s

How to distribute the processes?
How to manage the data flow?
How to recover processes when they crash?
How to monitor the whole system?
……





1 TB/s

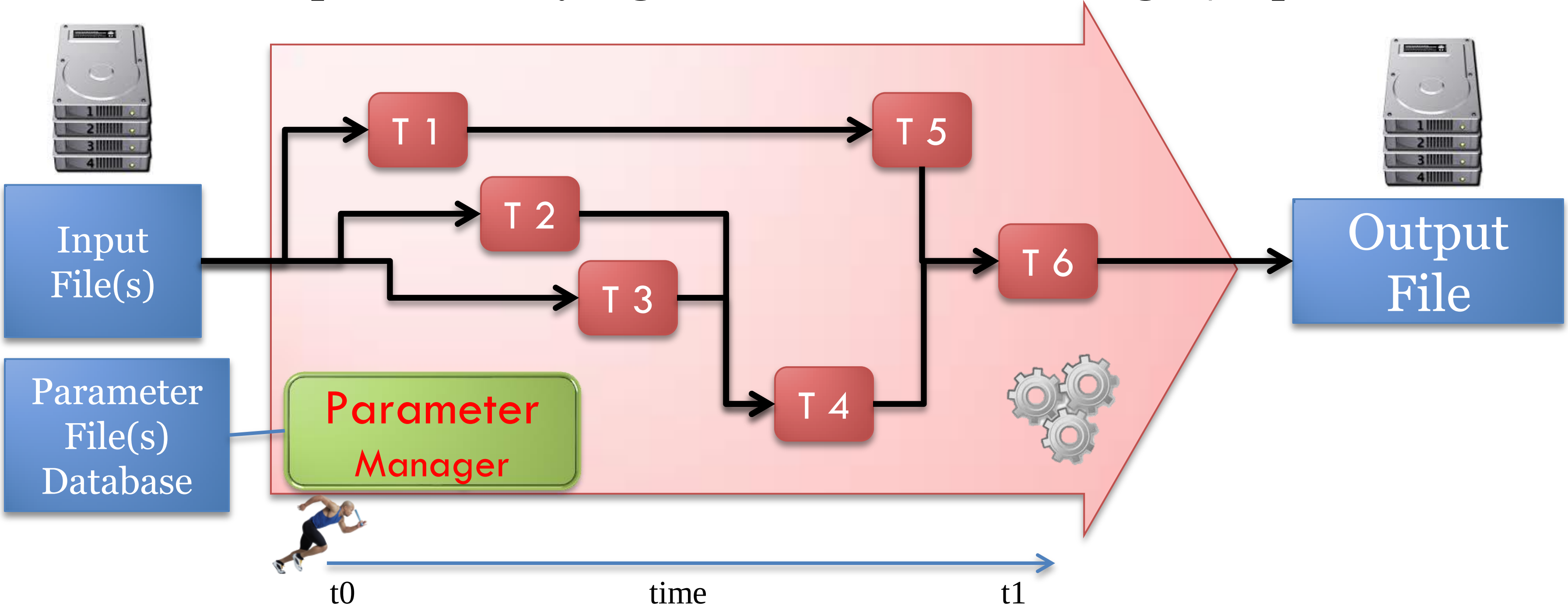


> 60 000
CPU-core
or Equivalent
GPU, FPGA, …

1 GB/s

FairRoot: Where we are now?

- ROOT event loop
- User code in Task hierarchy
- Task hierarchy runs sequentially in one process
- Tasks implement only algorithms (can be exchanged/replaced)



FairRoot: How to scale

- Computer have more and more cores.
  - Online clusters of CBM and Panda will have about 60.000 cores
  - One monolithic program only use one of this cores
  - How we can better use the computing power of the modern computers?
- C and C++ do not offer any support for concurrency!
- Embarrassingly parallel workload (Start as many FairRoot processes as cores are available)
  - Memory needed for each process ➔ expensive
  - How this scheme should work for the Online cluster?

Design constrains

- Highly flexible:
  - different data paths should be modeled.
- Adaptive:
  - Sub-systems are continuously under development and improvement
- Should work for simulated and real data:
  - developing and debugging the algorithms
- It should support all possible hardware where the algorithms could run (CPU, GPU, FPGA)
- It has to scale to any size! With minimum or ideally no effort.

Multi-processing vs. Multi-threading

- Different processes are insulated from each other by the OS, an error in one process cannot bring down another process.
- Inter-process communication can be used across network

- Error in one thread can bring down all the threads in the process.
- Inter-thread communication is fast

The best would be to find the correct balance between reliability and performance

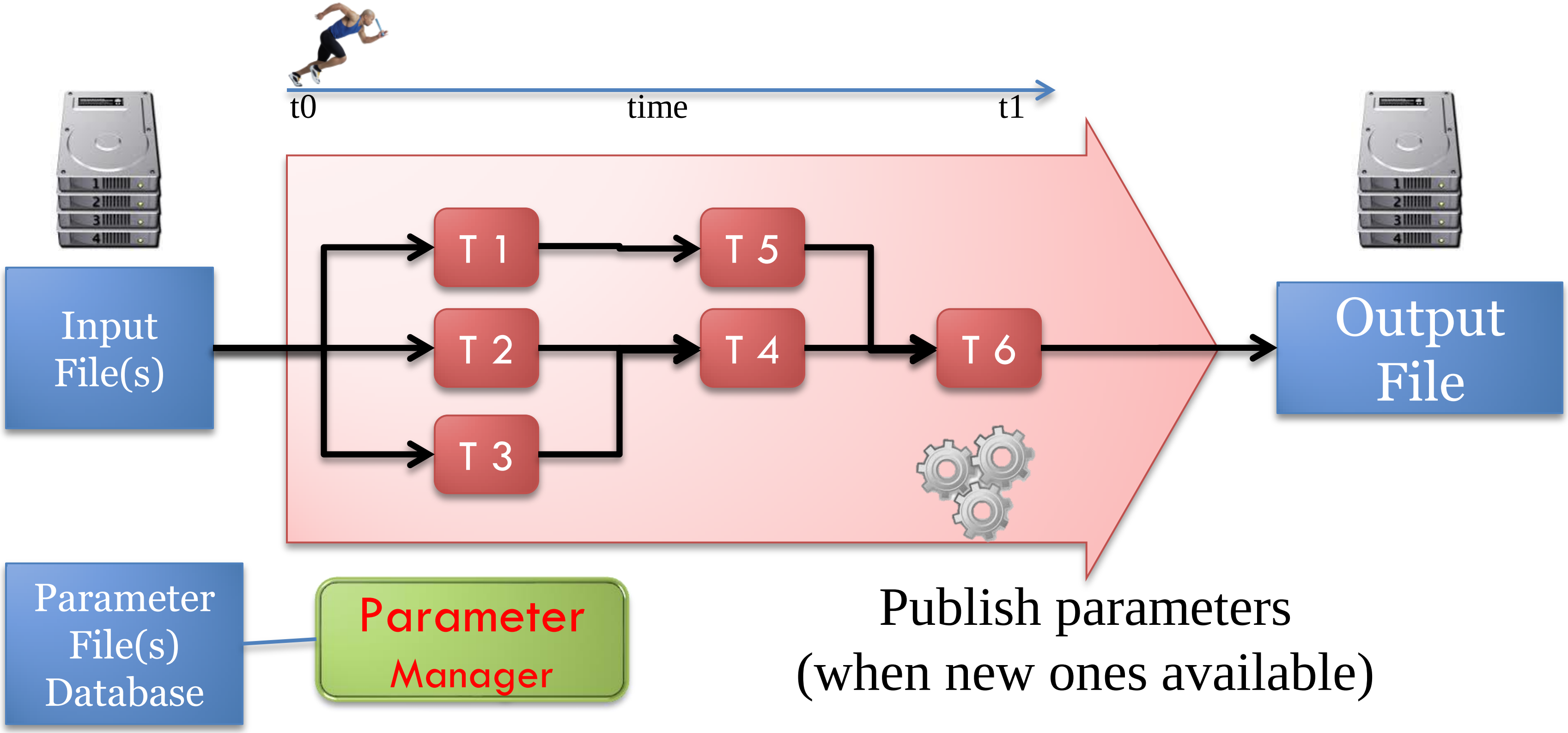
- Multi-process concept with message queues for data exchange

  - Each "Task" is a separate process, which can be also multithreaded, and the data exchange between the different tasks is done via messages.

  - Different topologies of tasks that can be adapted to the problem itself, and the hardware capabilities.

FairRoot: Where we are going ? (almost there!)

- Each Task is a process (can be Multi-threaded)
- Message Queues for data exchange
- Support multi-core and multi node



Publish parameters
(when new ones available)

Before Re-inventing the Wheel

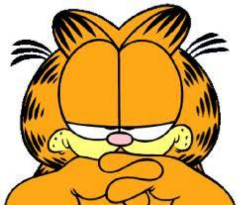
- What is available on the market and in the community?
  - A very promising package: ZeroMQ is available since 2011
- Do we intend to separate online and offline? NO
- Multithreaded concept or a message queue based one?
  - Message based systems allow us to decouple producers from consumers.
  - We can spread the work to be done over several processes and machines.
  - We can manage/upgrade/move around programs (processes) independently of each other.

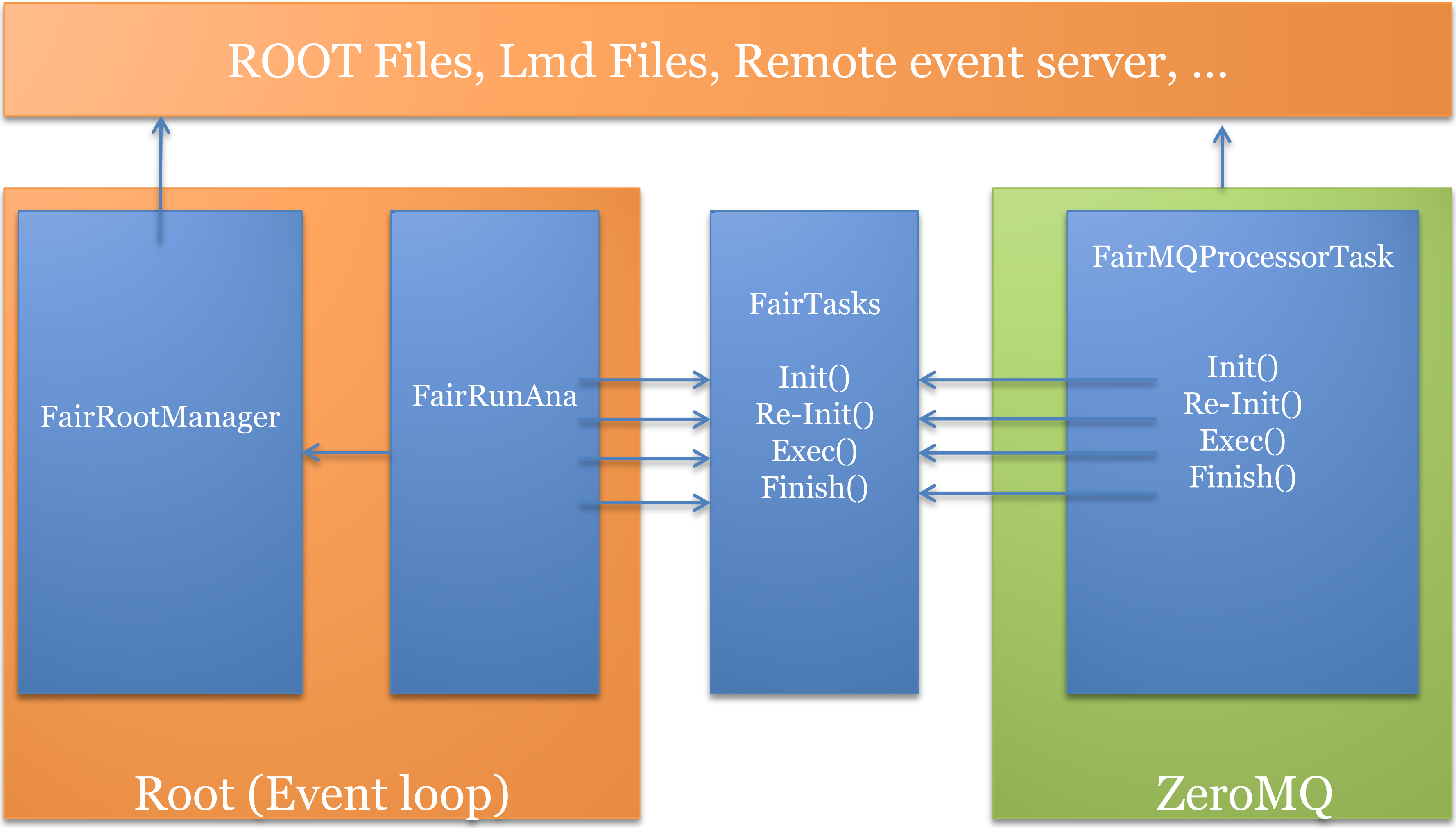
- A messaging library, which allows you to design a complex communication system without much effort
- Abstraction on higher level than MPI (programming model is easier)
- Is suitable for loosely coupled and more general distributed systems
- Multiplatform, multi-language (+30)
- Small (20K lines of C++ code)
- Large and active open source community.
- Open source LGPL free software (large community)

Current Status

- The Framework delivers some components which can be connected to each other in order to construct a processing pipeline(s).
- All components share a common base called Device (ZeroMQ Class).
- Devices are grouped by three categories:
  - Source:
    - Data Sampler
  - Message-based Processor:
    - Sink, Splitter, Merger, Buffer, Proxy
  - Content-based Processor:
    - Processor

Integrating the existing software:





ROOT Files, Lmd Files, Remote event server, …

FairRootManager

FairRunAna

FairTasks

Init()
Re-Init()
Exec()
Finish()

FairMQProcessorTask

Init()
Re-Init()
Exec()
Finish()

Root (Event loop)

ZeroMQ

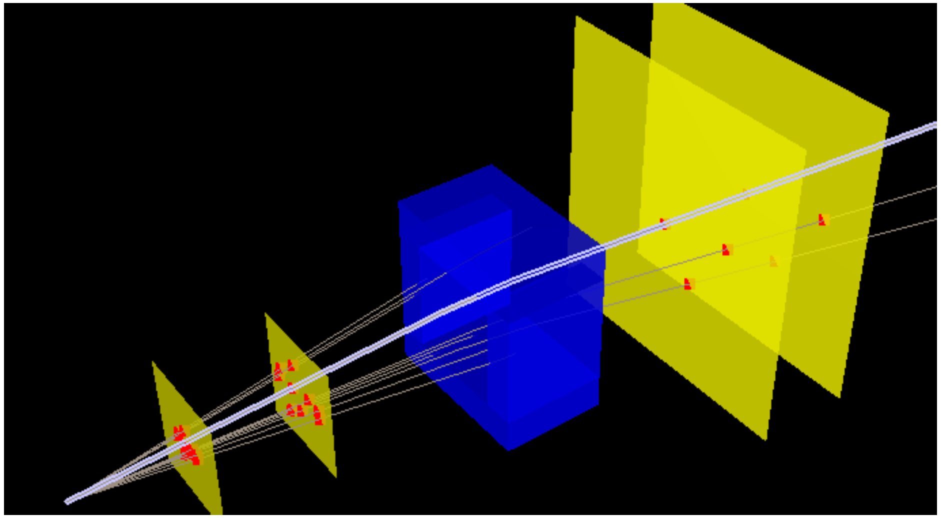
FairRoot: Example 3

4 -Tracking stations with a dipole field

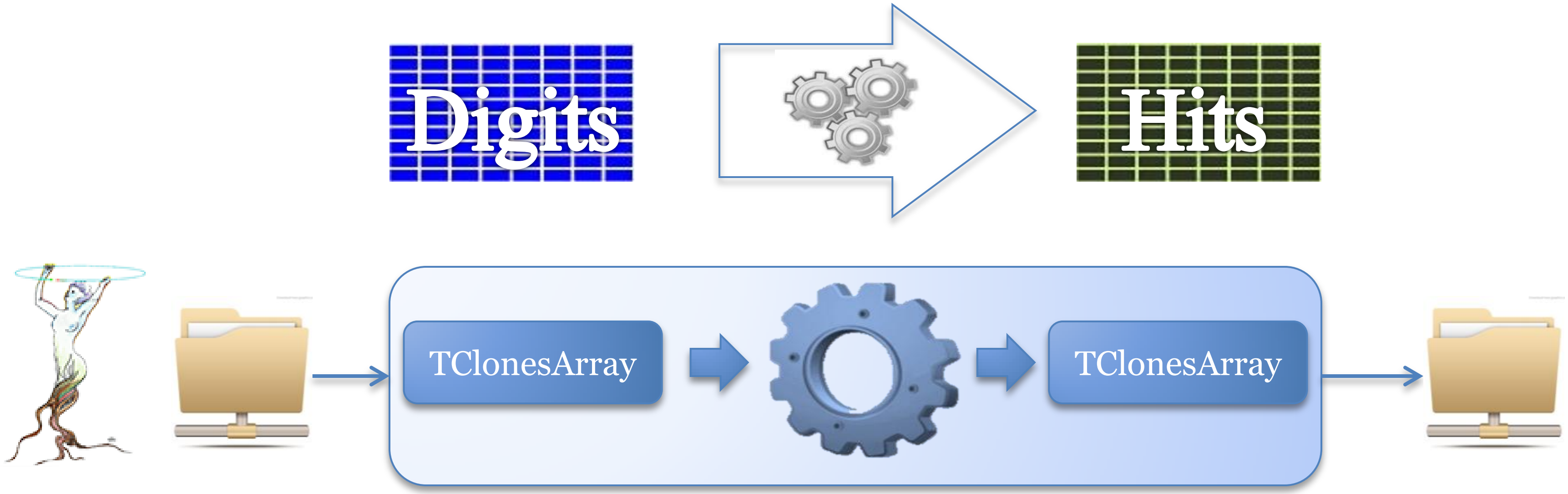
Simulation:
10k event: 300 Protons/ev

Digitization

Reconstruction:
Hit/Cluster Finder

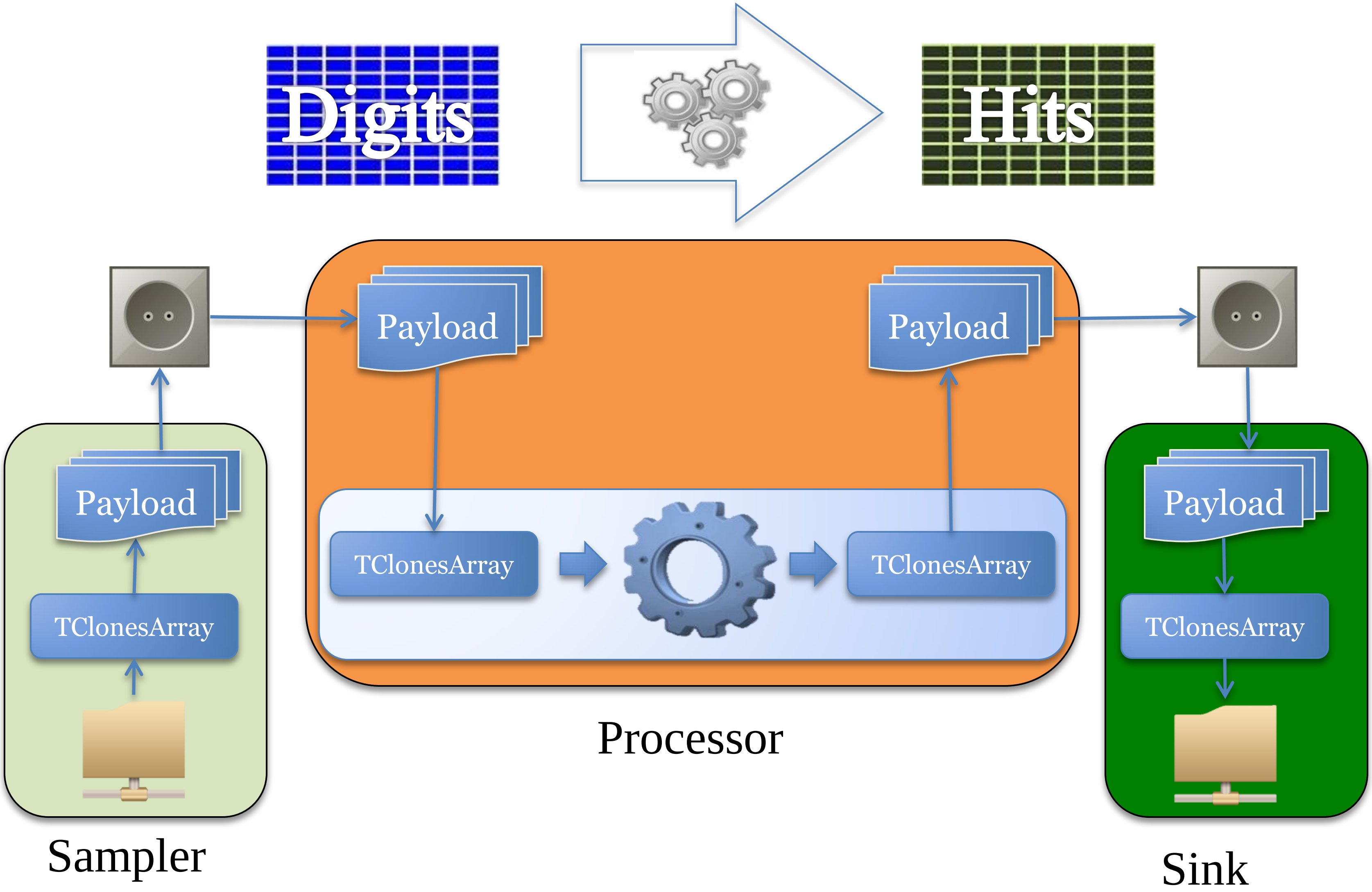


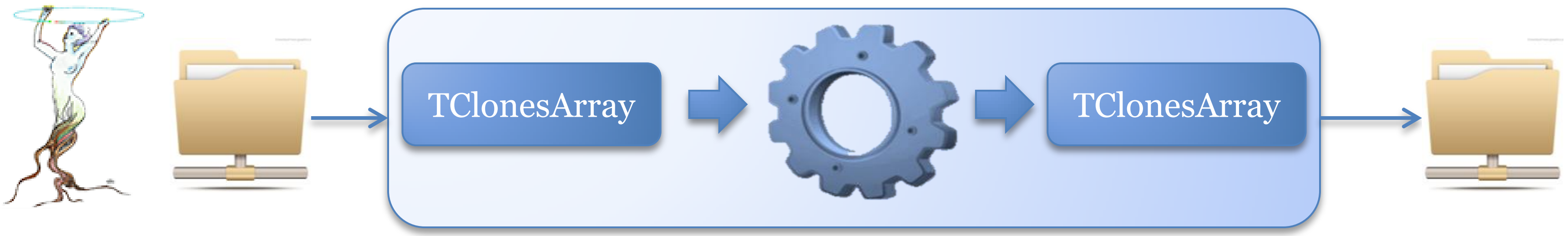
From digits to hits with ROOT:



RUN	CPU Time (s) (Wall time)	Memory (Mbyte)
10k Events, 300 Protons/event	100	263

From digits to hits with ØMQ:

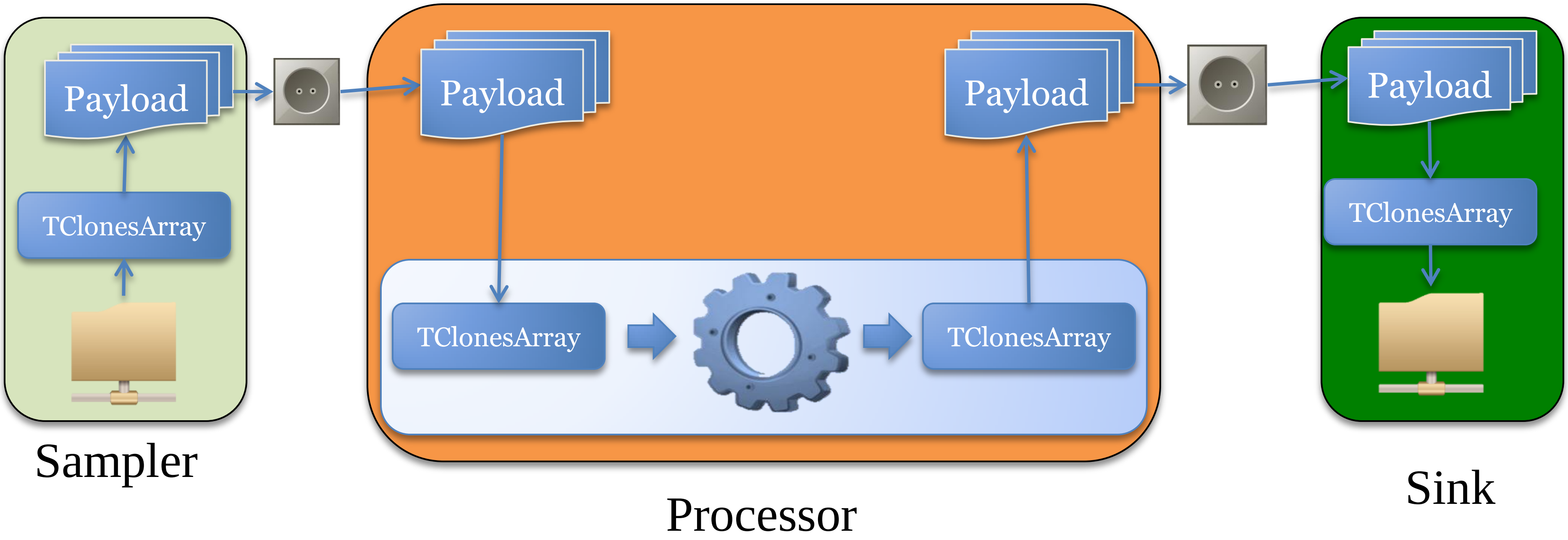




TClonesArray

TClonesArray

Vs.



Payload

TClonesArray

Payload

TClonesArray

TClonesArray

Payload

Payload

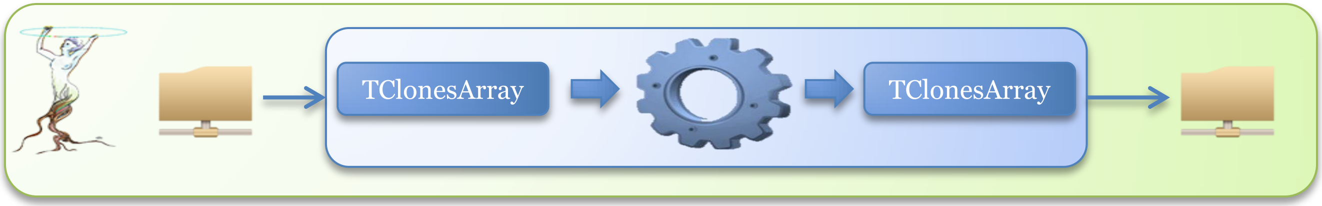
TClonesArray

Sampler

Processor

Sink

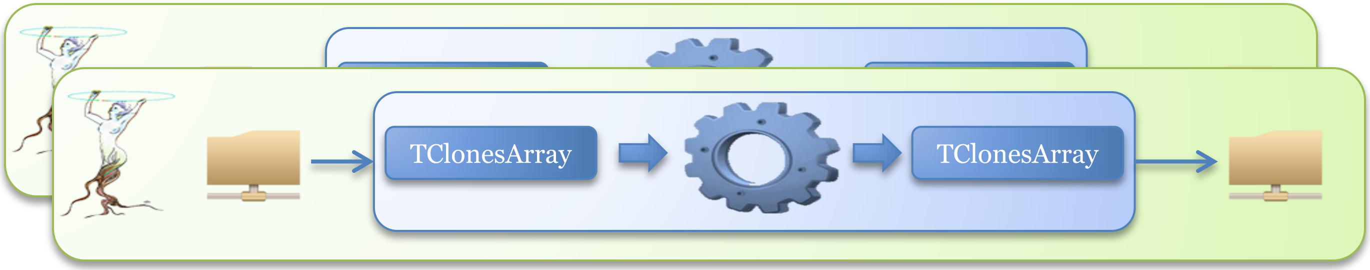
2 x 2.4 Xeon Quad core Intel Xeon
16 GB Memory



100 s
263 MB

Throughput ~ 1000 ev/s
Total Memory 263 Mb

2 x 2.4 Xeon Quad core Intel Xeon
16 GB Memory

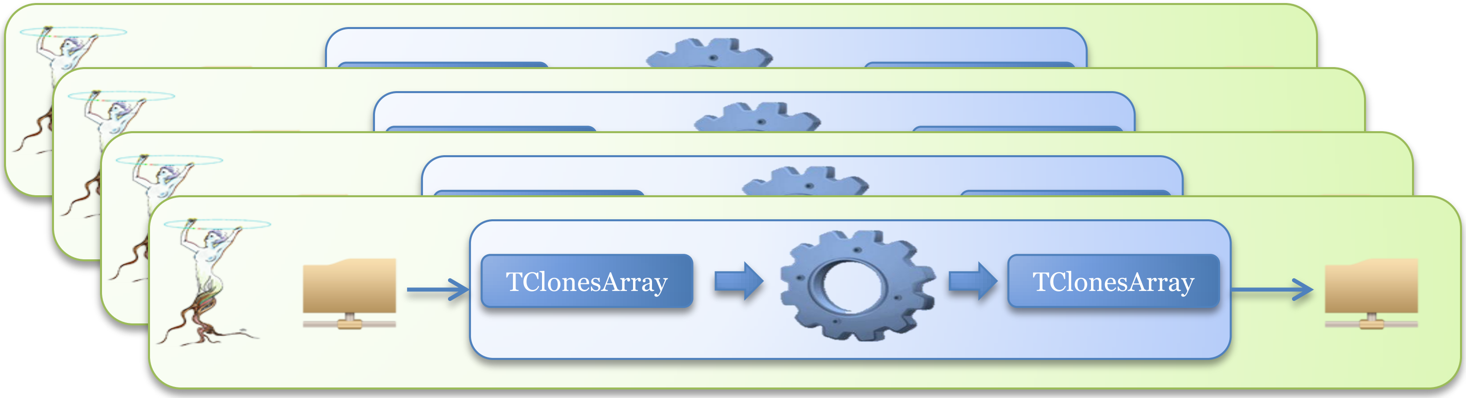


100 s

2* 263 MB

Throughput ~ 2000 ev/s
Total Memory 526 Mb

2 x 2.4 Xeon Quad core Intel Xeon
16 GB Memory



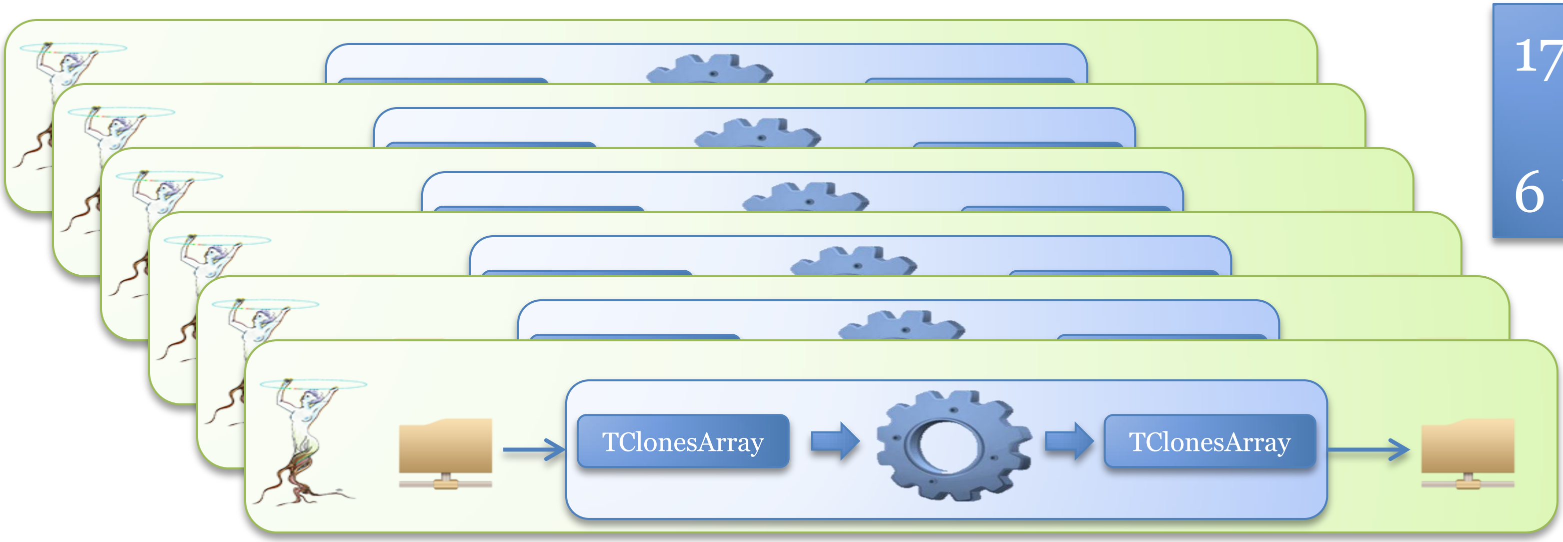
121 s

4* 263 Mb

Throughput ~ 3300 ev/s

Total Memory 1052 Mb

2 x 2.4 Xeon Quad core Intel Xeon
16 GB Memory

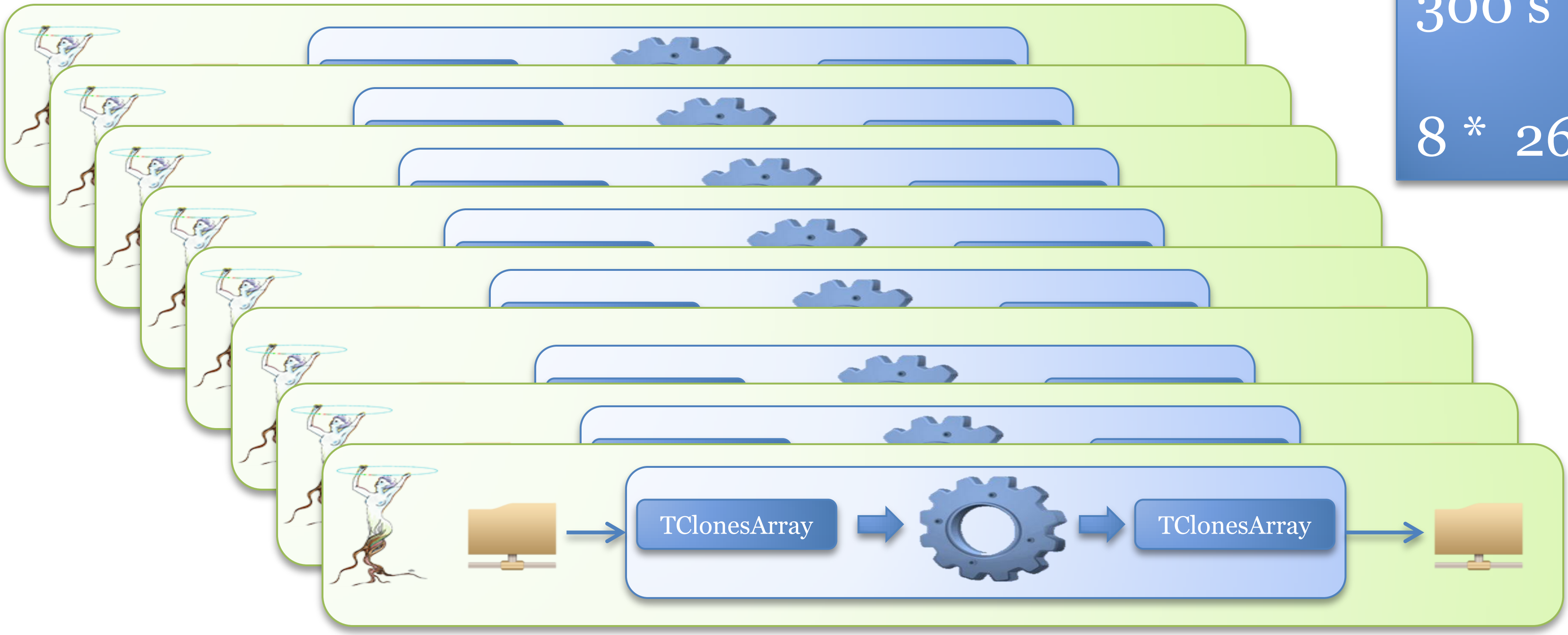


171 s

6 * 263 MB

Throughput ~ 3500 ev/s

2 x 2.4 Xeon Quad core Intel Xeon
16 GB Memory

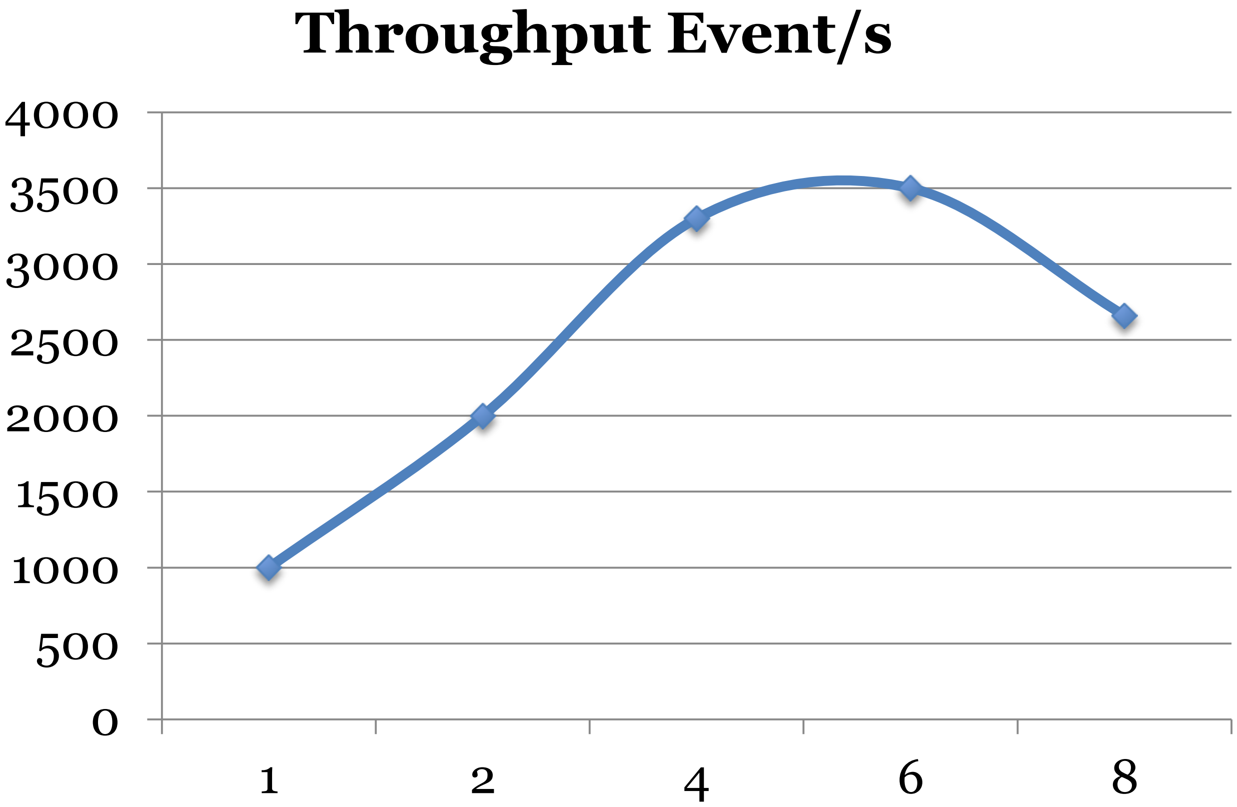


300 s

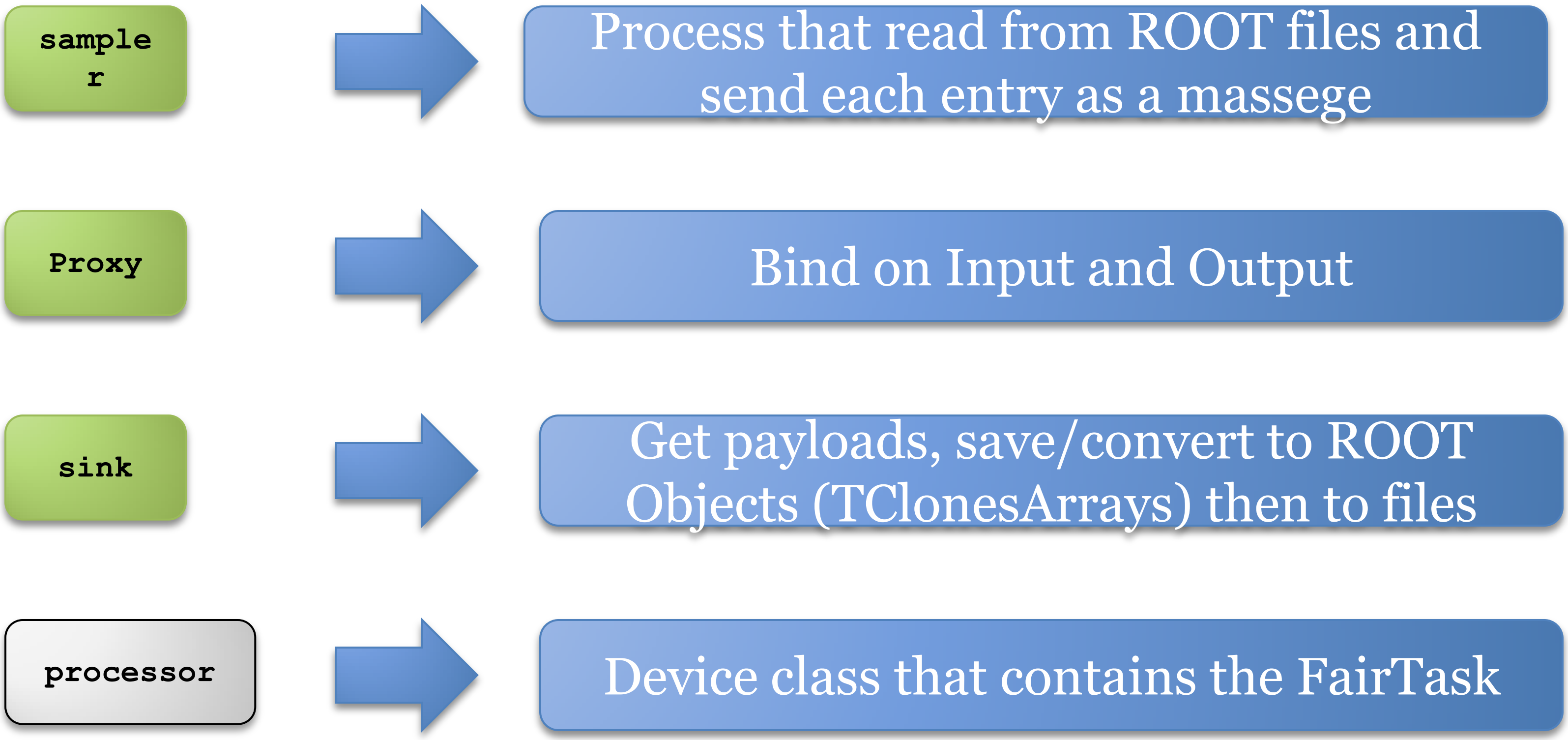
8 * 263 MB

Throughput ~ 2660 ev/s

2 x 2.4 Xeon Quad core Intel Xeon
16 GB Memory



Before we continue:



`sampler` → Process that read from ROOT files and send each entry as a massege

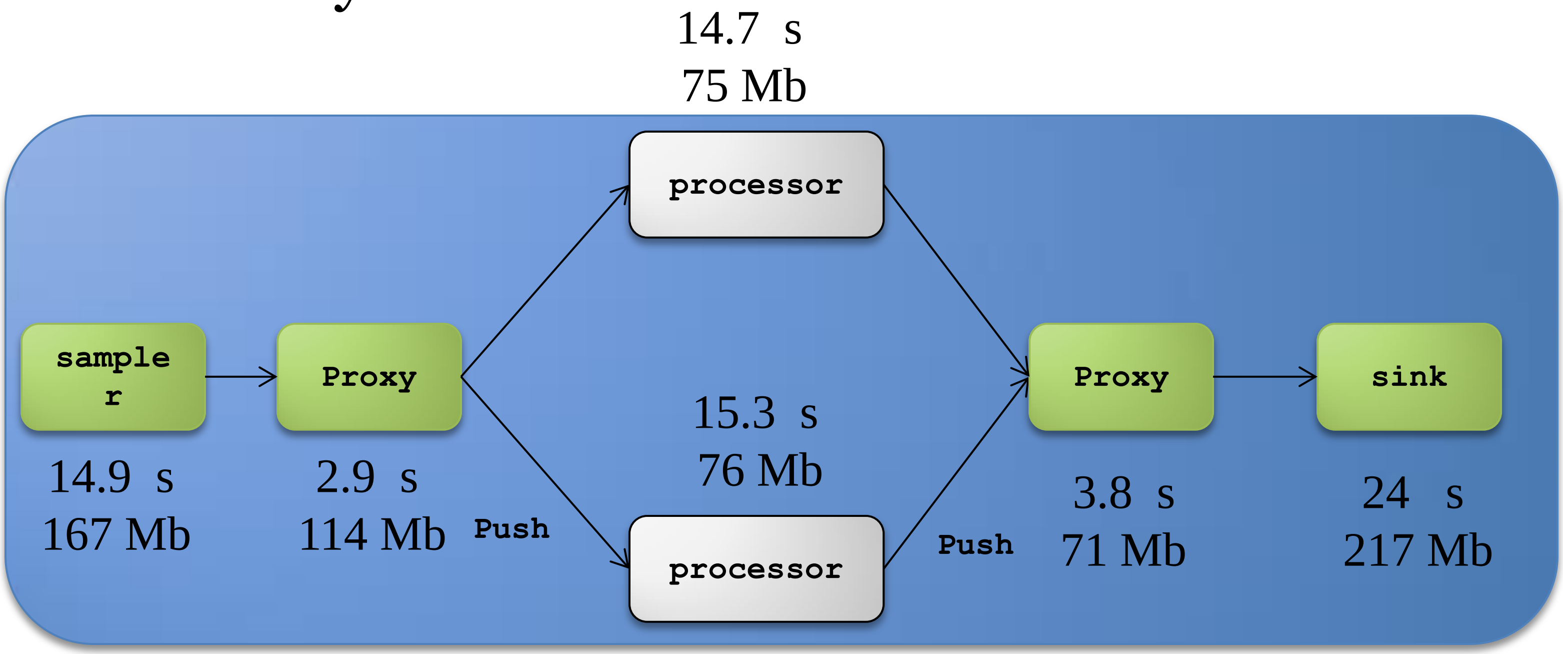
`Proxy` → Bind on Input and Output

`sink` → Get payloads, save/convert to ROOT Objects (TClonesArrays) then to files

`processor` → Device class that contains the FairTask

2 x 2.4 Xeon Quad core Intel Xeon
16 GB Memory

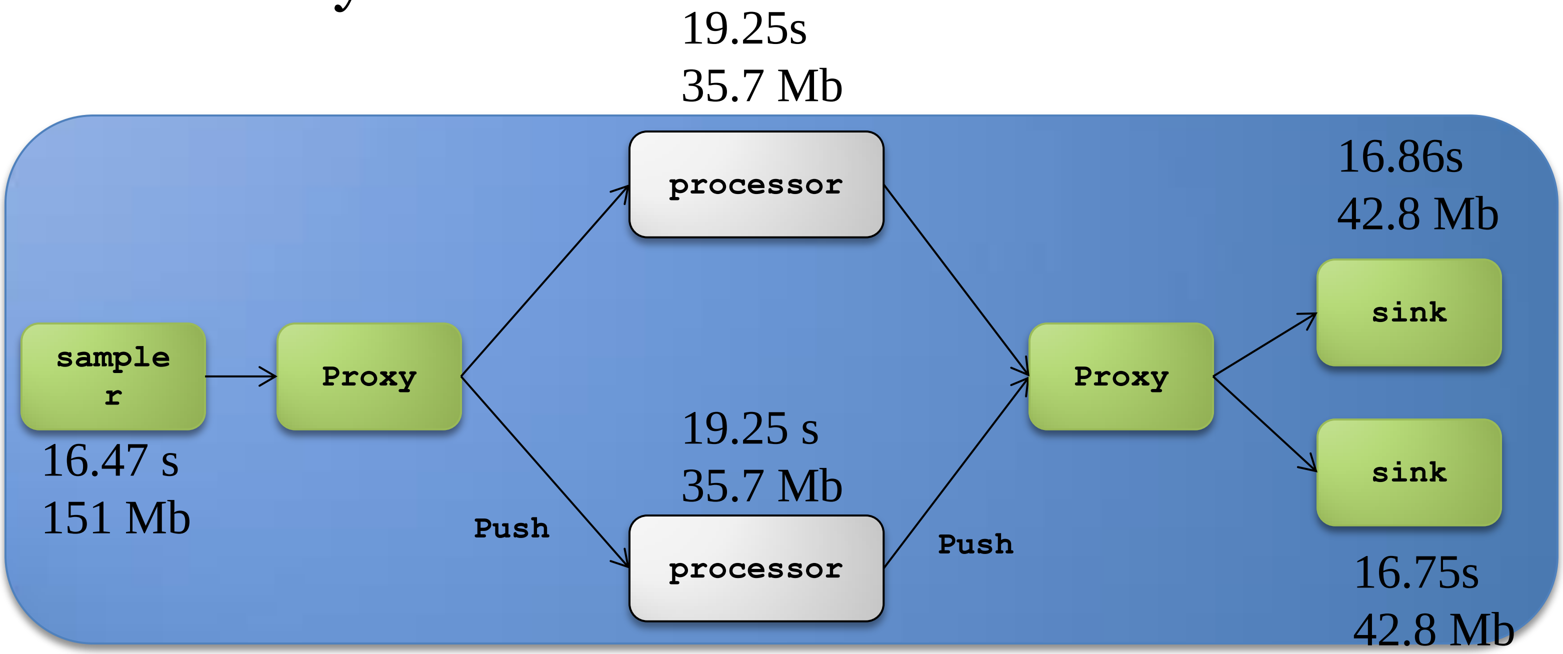
Wall time: 24 s



Throughput ~ 4166 ev/s
Total Memory 720 Mb

2 x 2.4 Xeon Quad core Intel Xeon
16 GB Memory

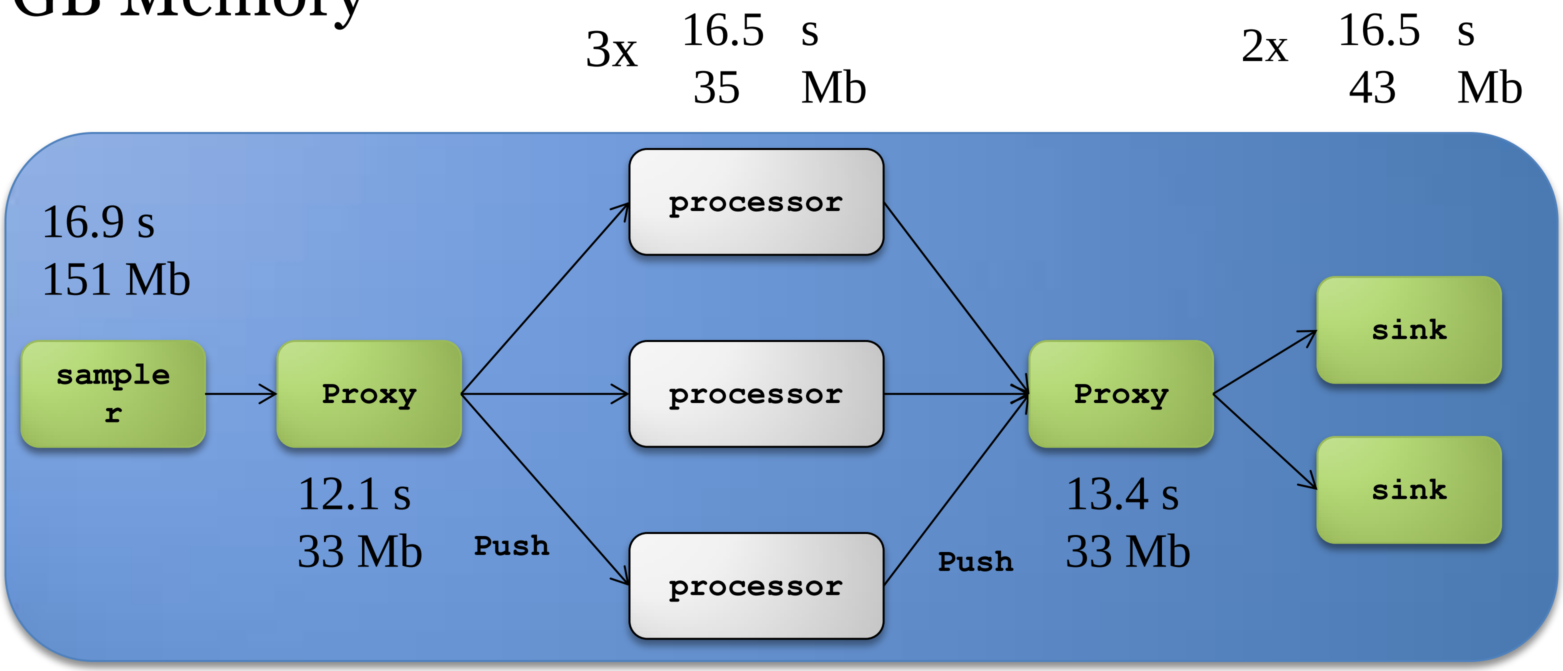
Wall time: 19.25 s



Throughput ~ 5190 ev/s
Total Memory 692 Mb

2 x 2.4 Xeon Quad core Intel Xeon
16 GB Memory

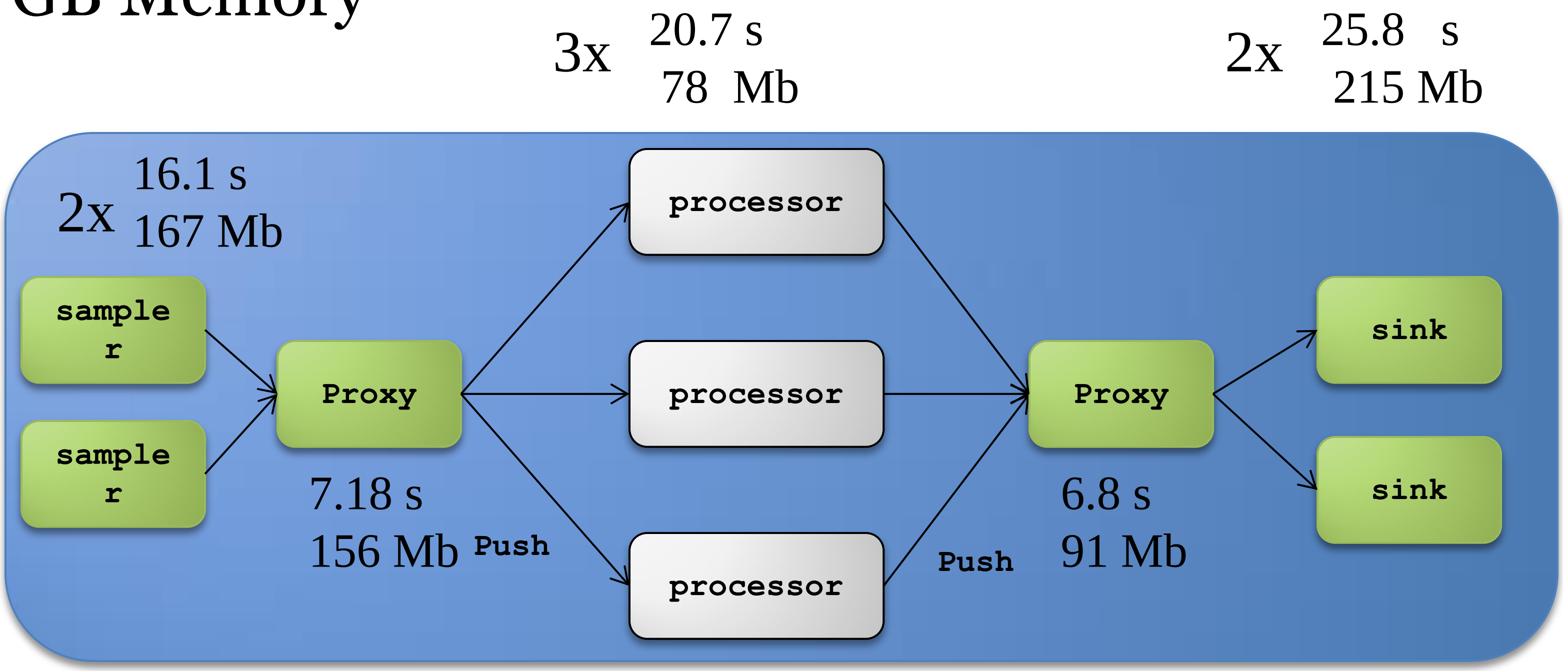
Wall time: 20.91 s



Throughput ~ 4780 ev/s
Total Memory 342 Mb

2 x 2.4 Xeon Quad core Intel Xeon
16 GB Memory

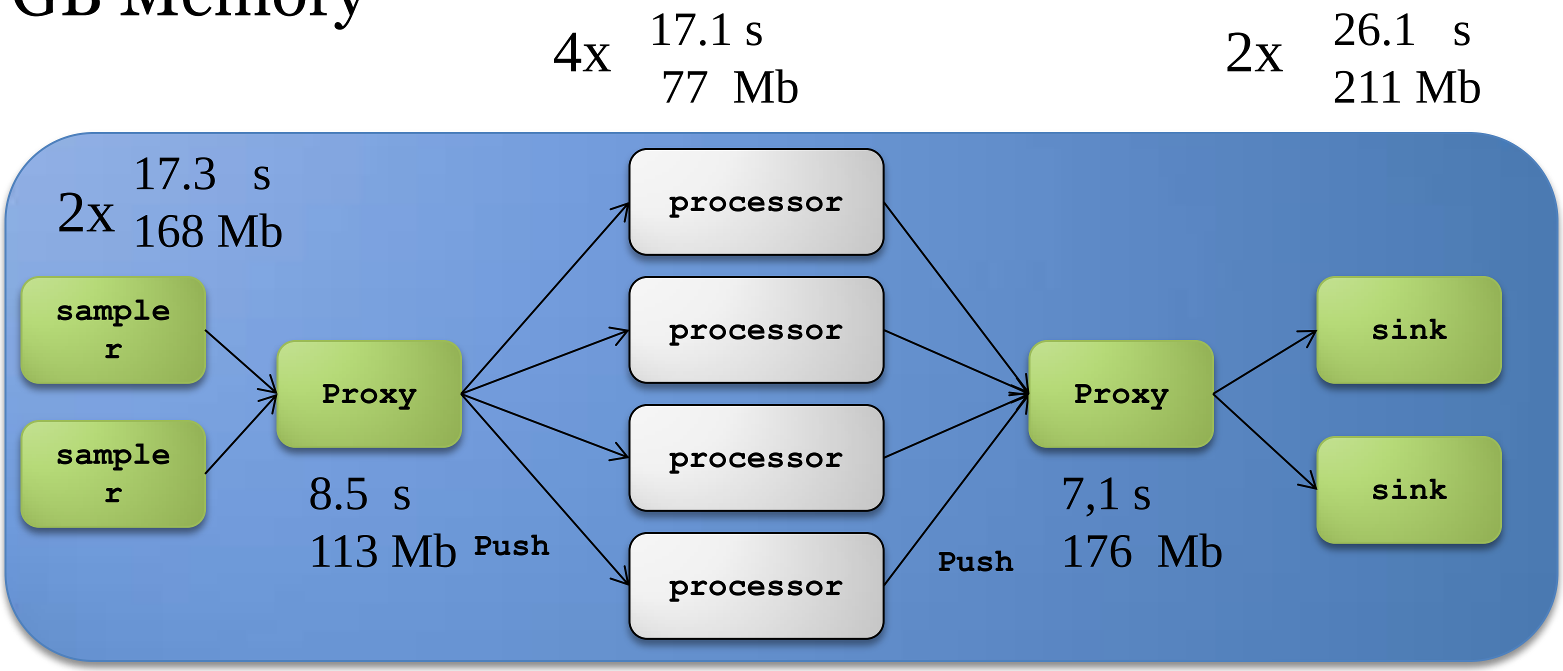
Wall time: 25.8 s



Throughput ~ 7320 ev/s
Total Memory 1245 Mb

2 x 2.4 Xeon Quad core Intel Xeon
16 GB Memory

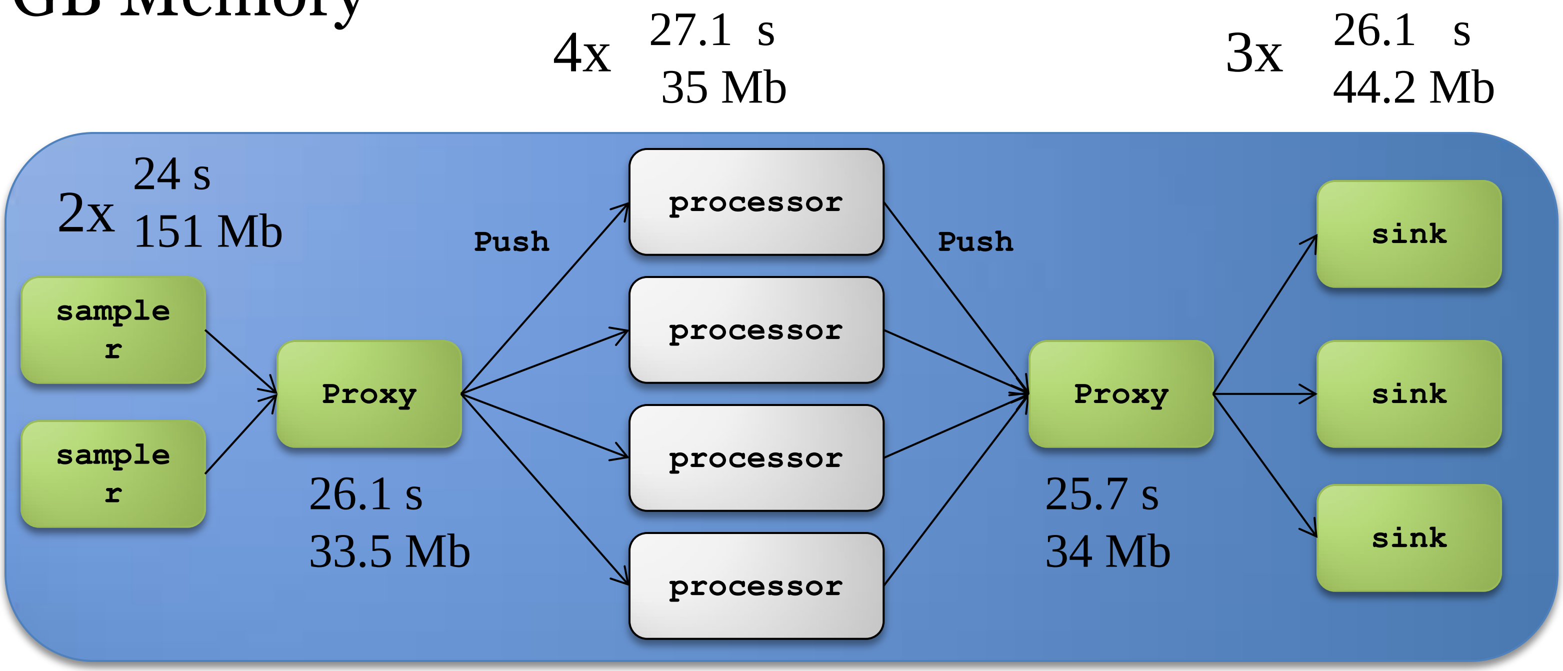
Wall time: 26.1 s



Throughput ~ 7400 ev/s
Total Memory 1355 Mb

Wall time: 30.5 s

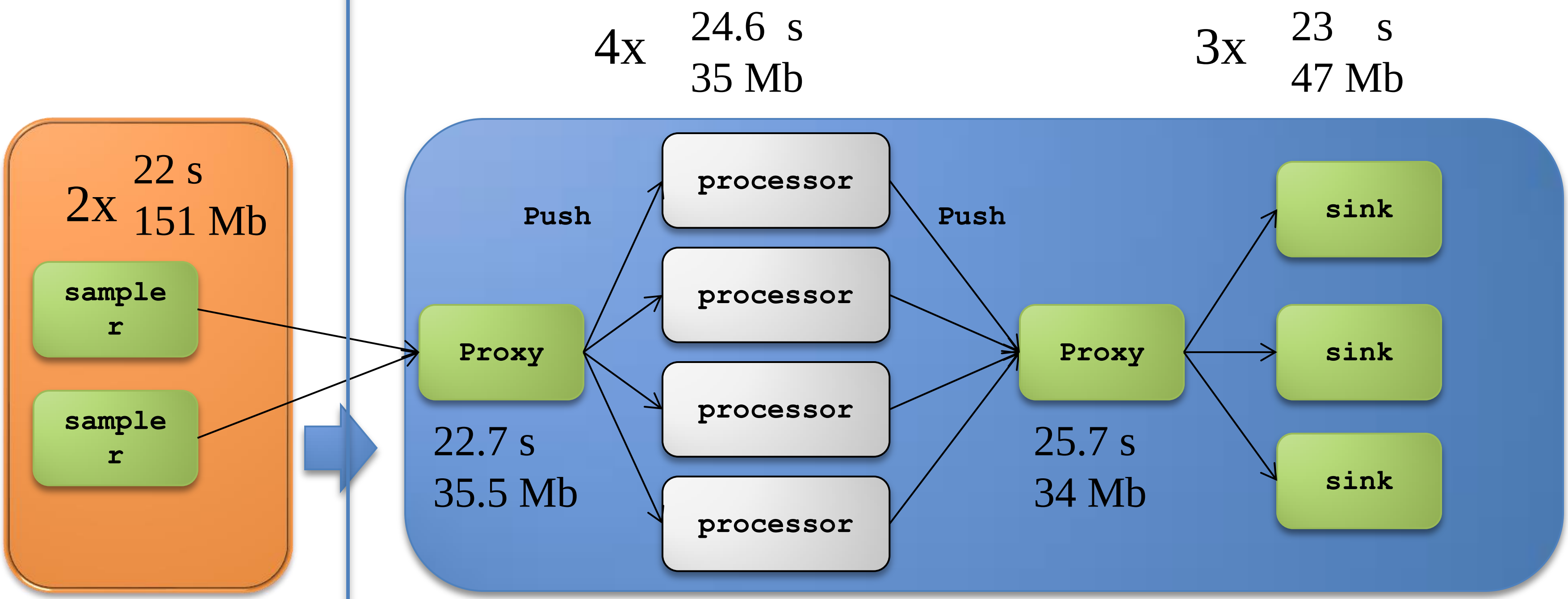
2 x 2.4 Xeon Quad core Intel Xeon
16 GB Memory



Throughput ~ 6560 ev/s
Total Memory 643 Mb

Wall time: 23.74 s

2 x 2.4 Xeon Quad core Intel Xeon
16 GB Memory



4x 24.6 s 35 Mb

3x 23 s 47 Mb

2x 22 s 151 Mb

sampler

sampler

Push

processor

processor

processor

processor

Push

Proxy

22.7 s
35.5 Mb

Proxy

25.7 s
34 Mb

sink

sink

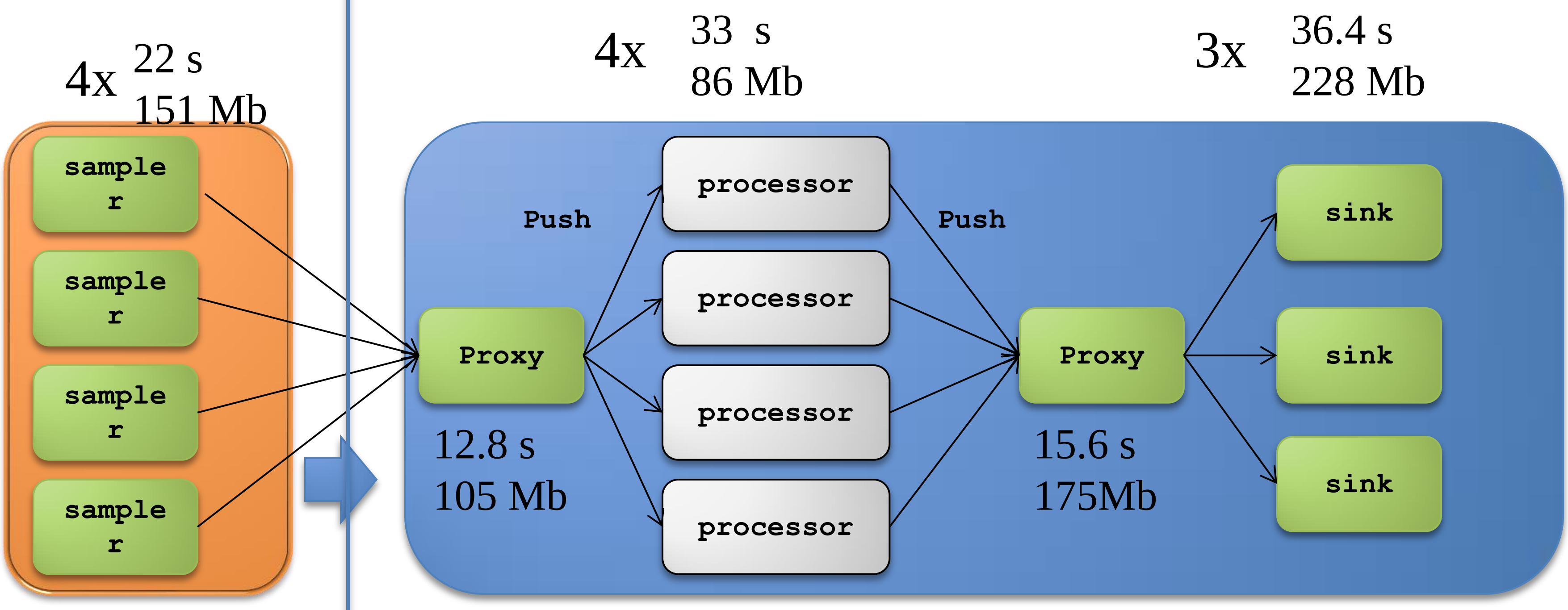
sink

Throughput ~ 8425 ev/s

Gigabit Ethernet

Wall time: 36.4 s

2 x 2.4 Xeon Quad core Intel Xeon
16 GB Memory



Throughput ~ 10990 ev/s

Gigabit
Ethernet

Summary

- ZeroMQ communication layer is integrated into our offline framework (FairRoot).
- On the short term we will keep both options: ROOT based event loop and concurrent processes communicating with each other via ZeroMQ.
- On long term we are moving away from single event loop to distributed processes.

Next Step: Design and development of a dynamic deployment system (DDS)

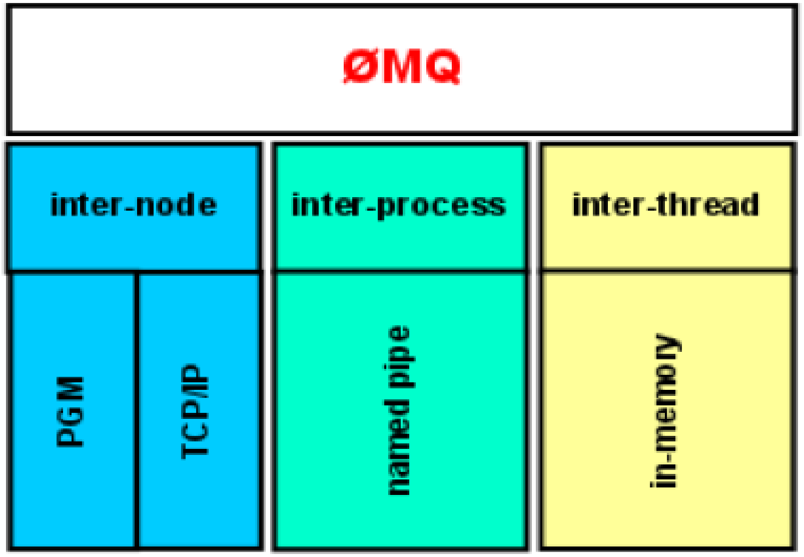
- STORM is very attractive but no native support for C++ !
- We need to utilize any RMS (Resource Management system)
- Support different topologies and process dependencies
- Device (process) is a single entity of the system
    - Each device has its own watchdog process
    - Devices are defined by a set of props and rules,
    - All devices are statically inherited (should support) 3 interfaces: IDDSConfig, IDDSStatus, and IDDSLog
- …..

Thank you

Backup

ZeroMQ sockets provide efficient transport options

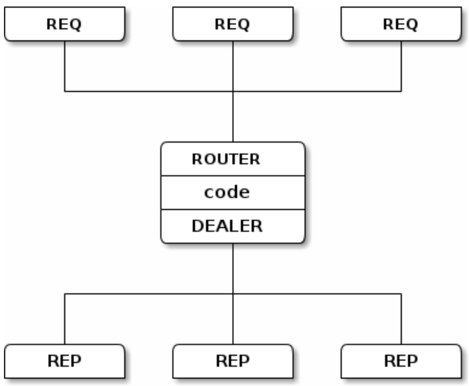
- Inter-thread
- Inter-process
- Inter-node
  - which is really just inter-process across nodes communication

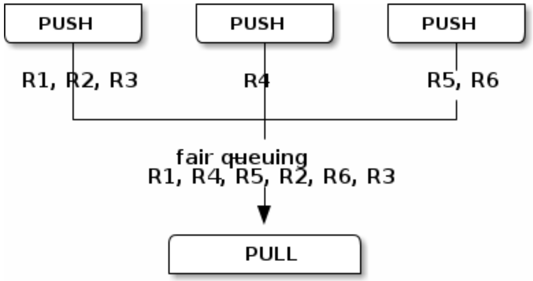


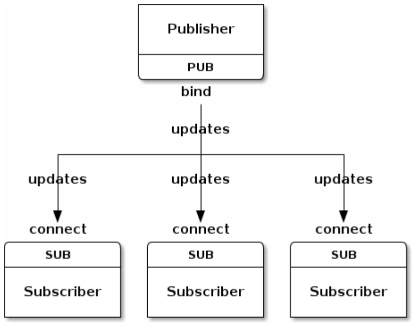
PMG : Pragmatic General Multicast (a reliable multicast protocol)
Named Pipe: Piece of random access memory (RAM) managed by the operating system and exposed to programs through a file descriptor and a named mount point in the file system. It behaves as a first in first out (FIFO) buffer

The built-in core ØMQ patterns are:

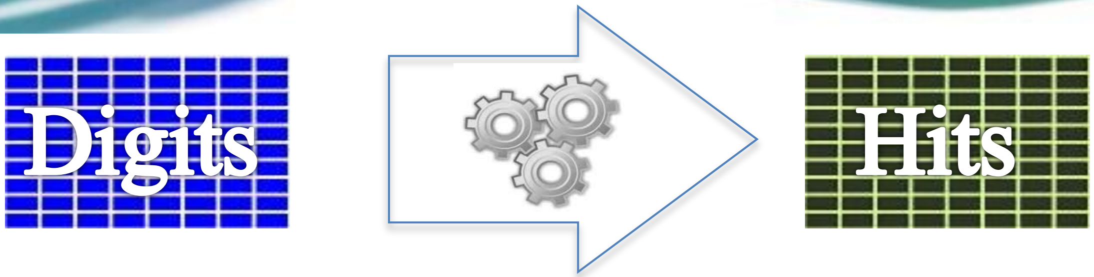
- **Request-reply**, which connects a set of clients to a set of services. (remote procedure call and task distribution pattern)
- **Publish-subscribe**, which connects a set of publishers to a set of subscribers. (data distribution pattern)
- **Pipeline**, which connects nodes in a fan-out / fan-in pattern that can have multiple steps, and loops. (Parallel task distribution and collection pattern)
- **Exclusive pair**, which connect two sockets exclusively

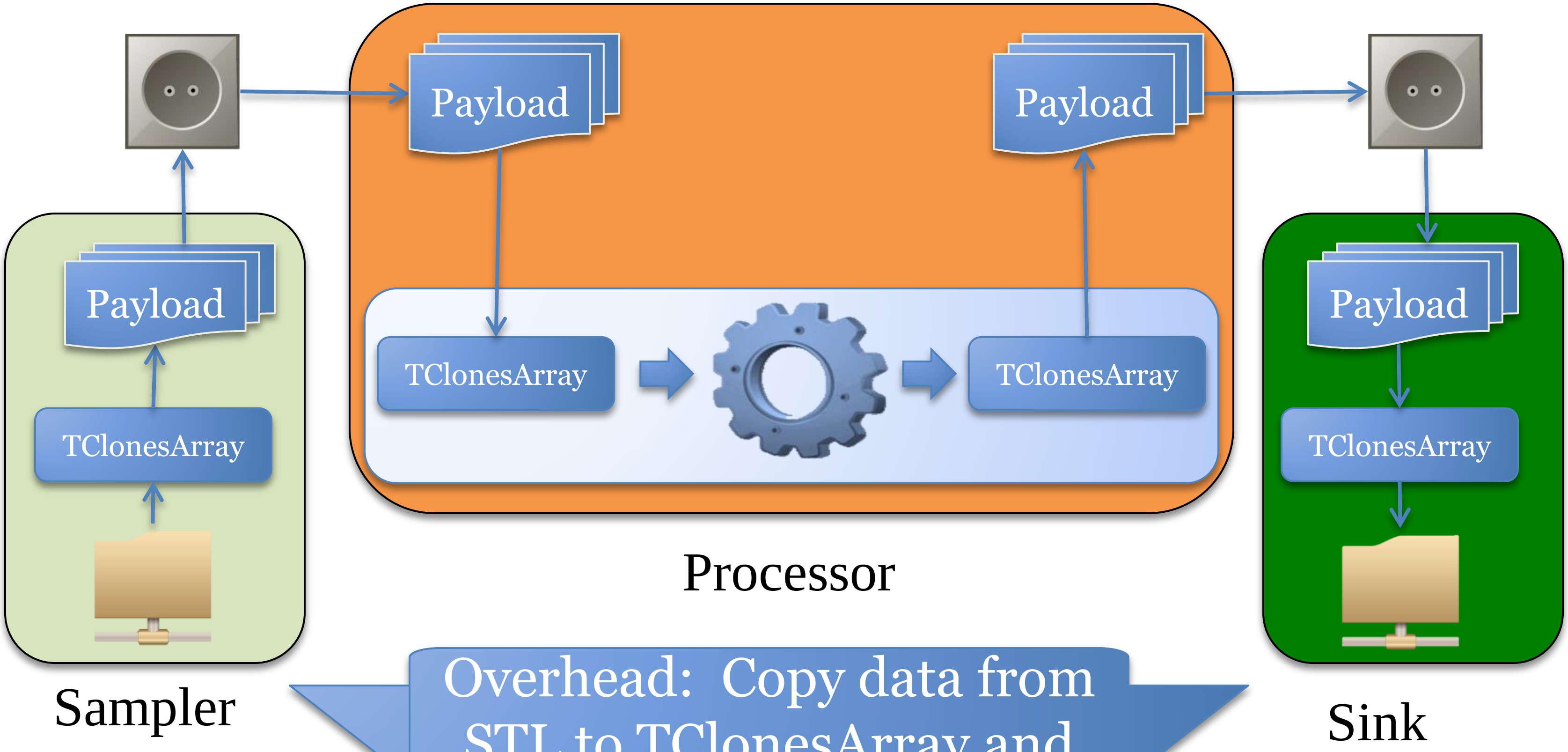






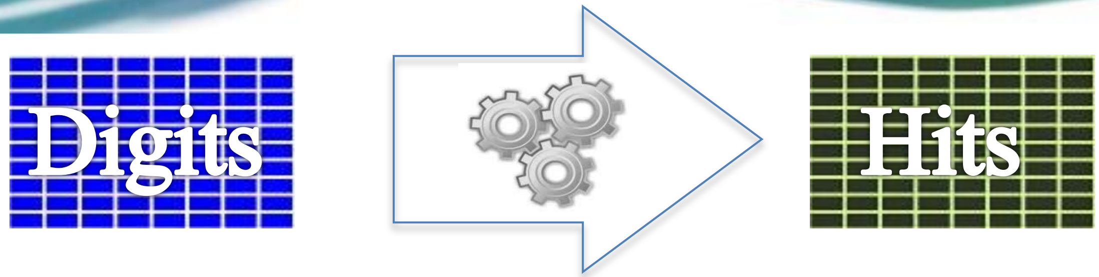
ØMQ

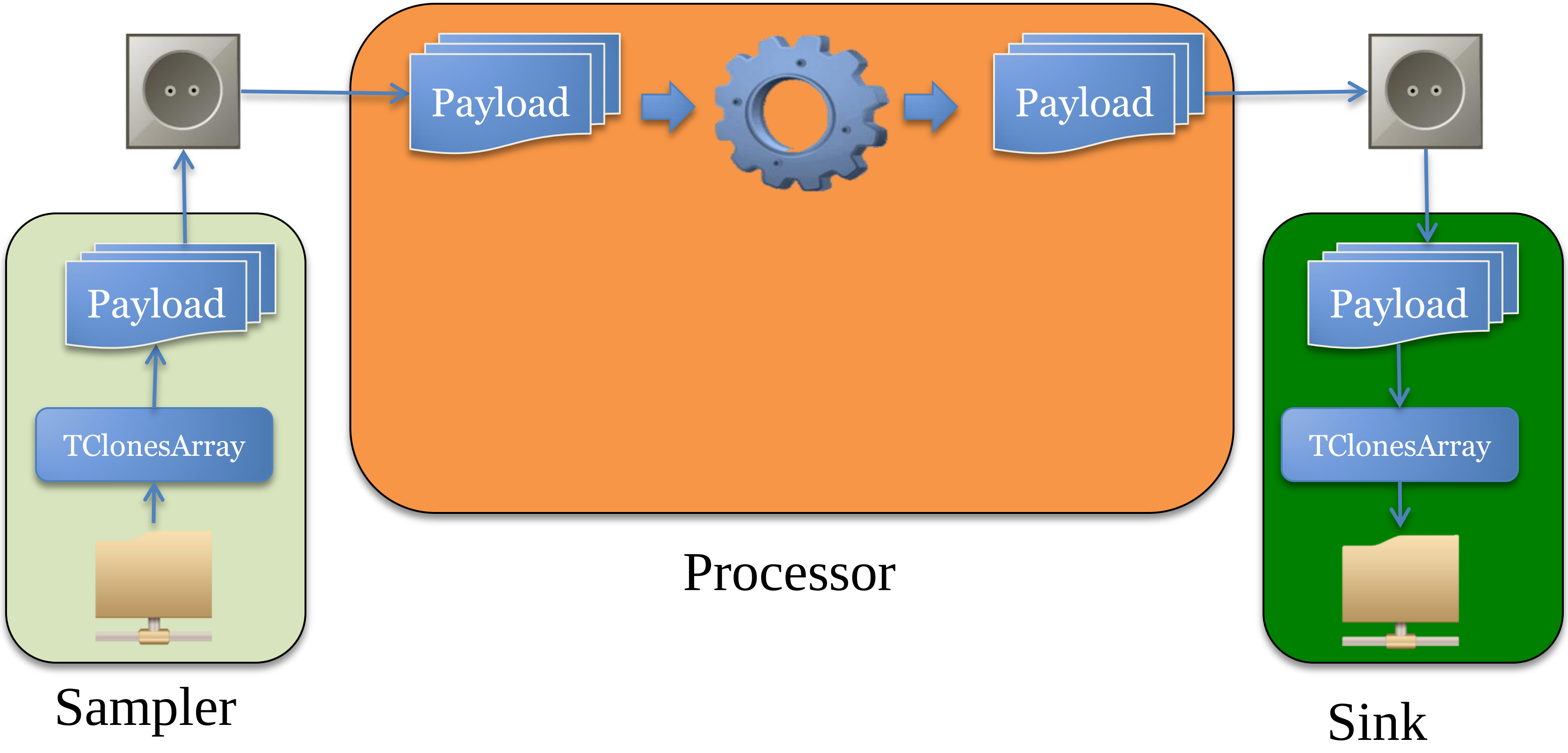




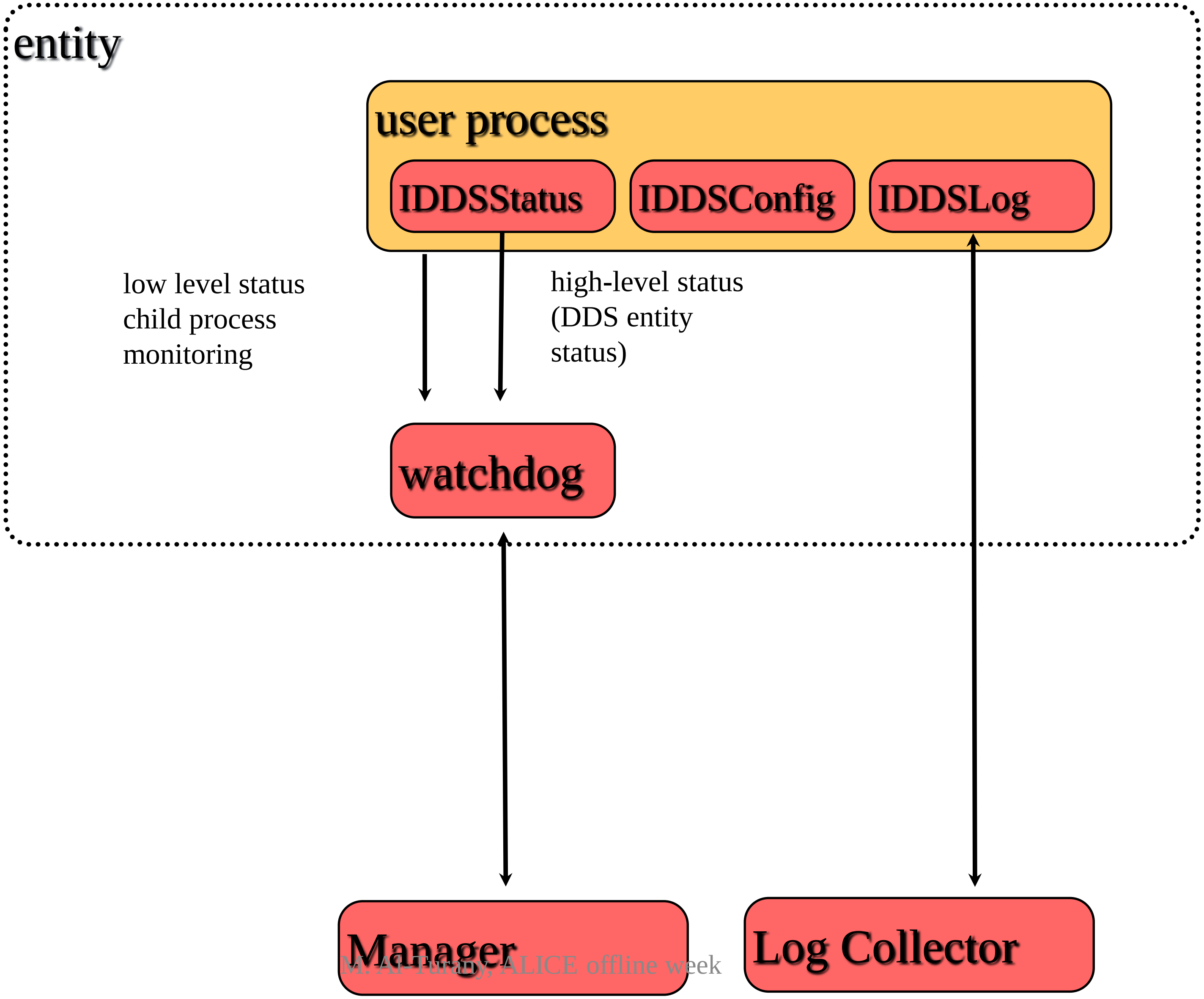
Overhead: Copy data from STL to TClonesArray and back

ØMQ


Digits
Hits


Payload
Payload
Payload
TClonesArray
Payload
TClonesArray
Processor
Sampler
Sink

Entities



entity

user process

IDDSStatus

IDDSConfig

IDDSLog

low level status
child process
monitoring

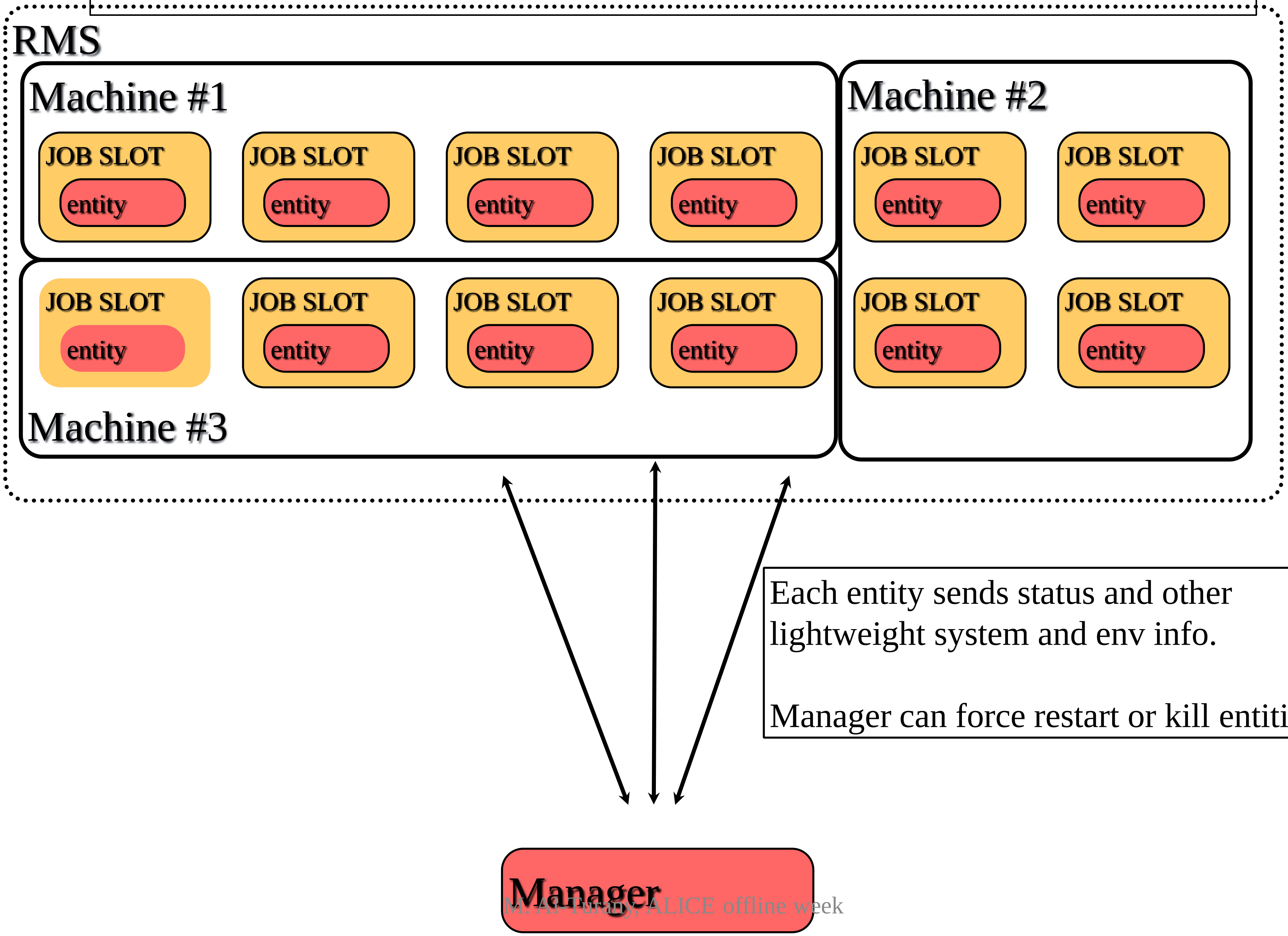
high-level status
(DDS entity
status)

watchdog

Manager

Log Collector

Entities



Each entity sends status and other lightweight system and env info.

Manager can force restart or kill entities