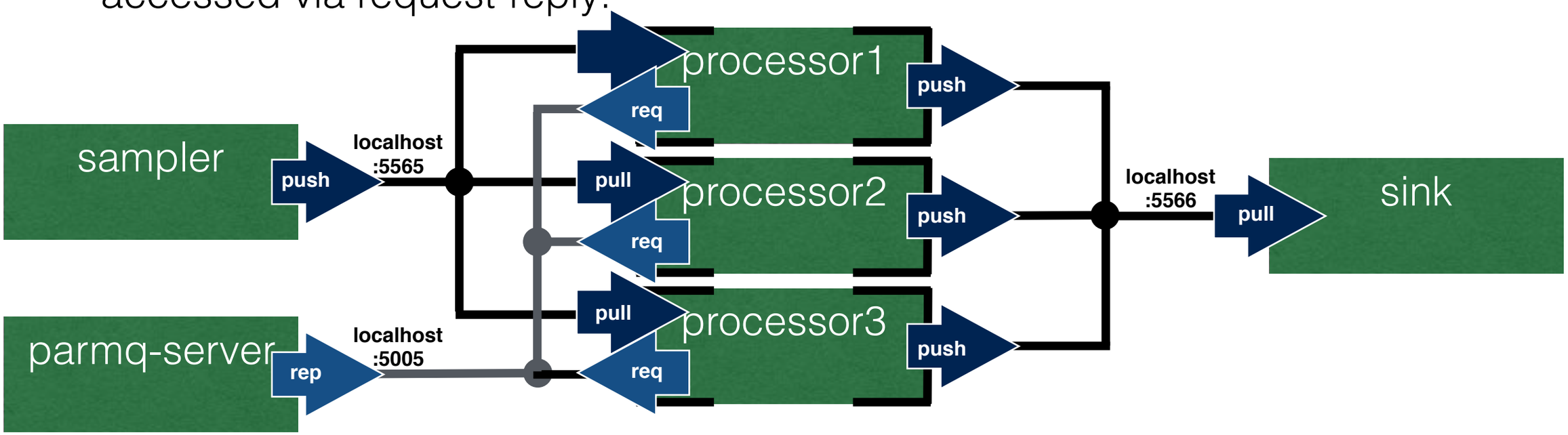
# Online Processing p.2

Radoslaw Karabowicz
Dennis Klein
FairRoot Group, GSI

https://github.com/FairRootGroup/FairRoot/tree/dev/examples/MQ/9-PixelDetector

# FairMQ topology

- defined in a JSON file;
- contains the list of devices (by the unique IDs);
- specifies connection types and communication patterns, addresses and ports, buffer sizes, verbosity level;
- example view of a topology with 1 sampler (bind), 1 sink (bind), 3 processors (connect) with push-pull communication pattern + a patameter server accessed via request-reply:



# FairMQ example 9

Radoslaw Karabowicz, GSI

https://github.com/FairRootGroup/FairRoot/tree/dev/examples/MQ/9-PixelDetector

# FairMQ example9

- provide simple example to show how to switch from single core to multicore pipeline processing;
- show how to use generic MQ tools;
- show how to send and receive FairRoot data;
- show how to execute the FairRoot tasks;

https://github.com/FairRootGroup/FairRoot/tree/dev/examples/MQ/9-PixelDetector

# FairMQ example9

- 3 stations with 4 rectangular sensors each:
  - size: 5x 5cm$^2$, inner hole: 1x1cm$^2$, at z= 5cm
  - size: 10x10cm$^2$, inner hole: 1x1cm$^2$, at z=10cm
  - size: 20x20cm$^2$, inner hole: 2x2cm$^2$, at z=20cm
- each sensor divided into pixels (0.01x0.01cm$^2$), that are grouped into FE modules (110pixels x 116pixels)

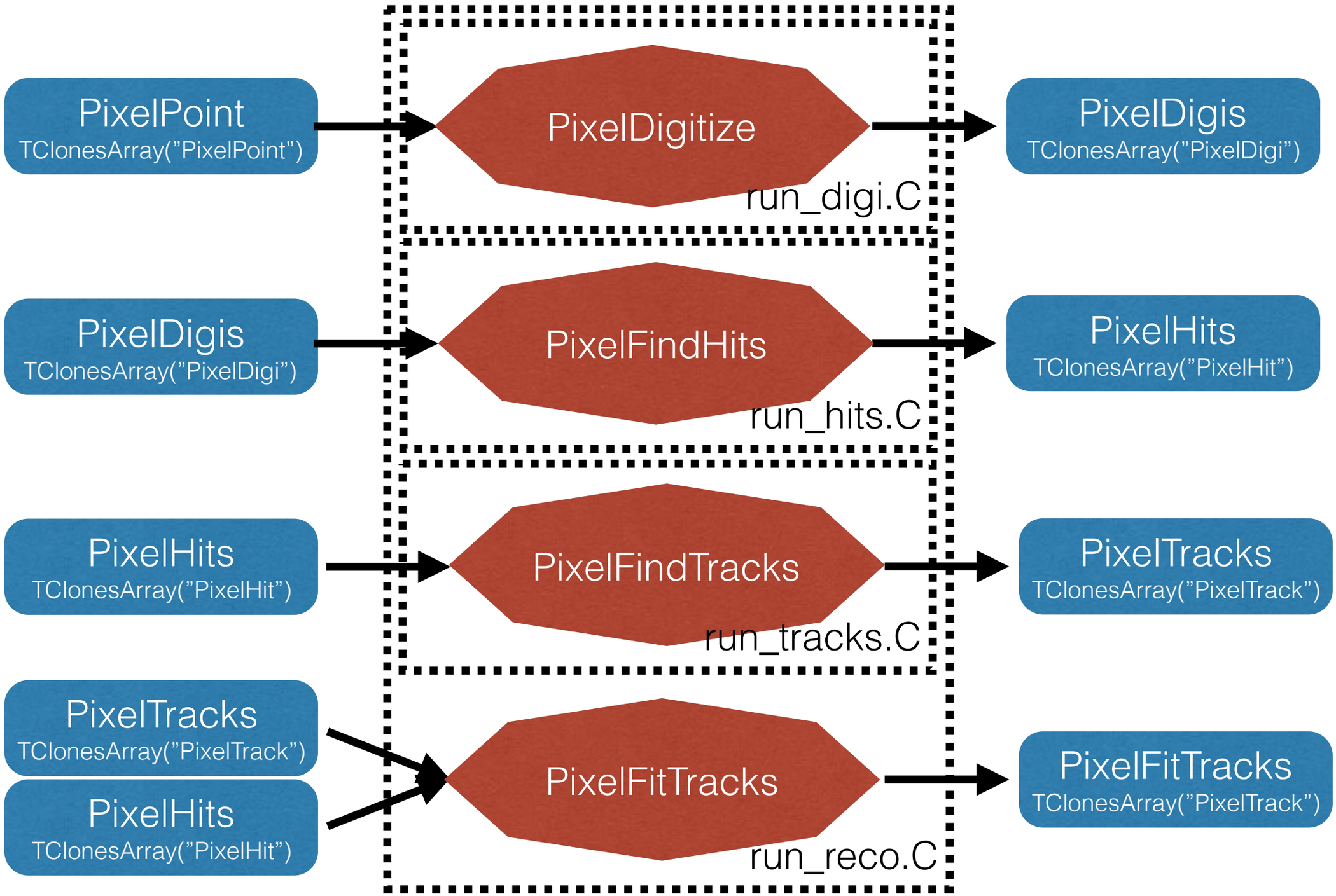
FEs numbering in one sensor

| | | | | | |
|---|---|---|---|---|---|
| … | | | | | |
| FE 5 | … | | | | |
| FE 4 | FE 68 | … | | | |
| FE 3 | FE 67 | FE 131 | … | | |
| FE 2 | FE 66 | FE 130 | FE 194 | … | |
| FE 1 | FE 65 | FE 129 | FE 193 | FE 257 | … |

```
[INFO   ] ------------ Pixel Digitizer : Summary --------
[INFO   ]  Events:        100000
[INFO   ]  MC Points:     944127    ( 9.44127 per event )
[INFO   ]  Digis:         940281    ( 9.40281 per event )
[INFO   ] ----------------------------------------------
[INFO   ] ------------ Pixel Hit Finder : Summary -------
[INFO   ]  Events:        100000
[INFO   ]  Digis:         940281    ( 9.40281 per event )
[INFO   ]  Hits:          940281    ( 9.40281 per event )
[INFO   ] ----------------------------------------------
[INFO   ] ------------ Pixel Track Finder : Summary -----
[INFO   ]  Events:        100000
[INFO   ]  Hits:          940281    ( 9.40281 per event )
[INFO   ]  Tracks:        281431    ( 2.81431 per event )
[INFO   ] ----------------------------------------------
[INFO   ] ------------ Pixel Track Fitter : Summary -----
[INFO   ]  Events:        100000
[INFO   ]  Tracks:        281431    ( 2.81431 per event )
[INFO   ]  Fitted Tracks: 281431    ( 2.81431 per event )
[INFO   ] ----------------------------------------------
```

https://github.com/FairRootGroup/FairRoot/tree/dev/examples/MQ/9-PixelDetector

# FairMQ example9



https://github.com/FairRootGroup/FairRoot/tree/dev/examples/MQ/9-PixelDetector

# MQ devices for data processing

- FairMQEx9Sampler reads data branch names specified by AddInputBranchName(string) from the input file(s) set by AddInputFileName(string);

- template <typename T> FairMQEx9TaskProcessor runs a task of class T. The class T needs to have the following functions:

```
void GetParList (TList* parList);
void InitMQ     (TList* parList);
void ExecMQ     (TList* inputList, TList* outputList);
```

- FairMQEx9FileSink creates output file set by SetOutputFileName(string) and tree, with the branches specified by AddOutputBranch(string className, string branchName). Currently only two classes (FairEventHeader and TClonesArray) are allowed:

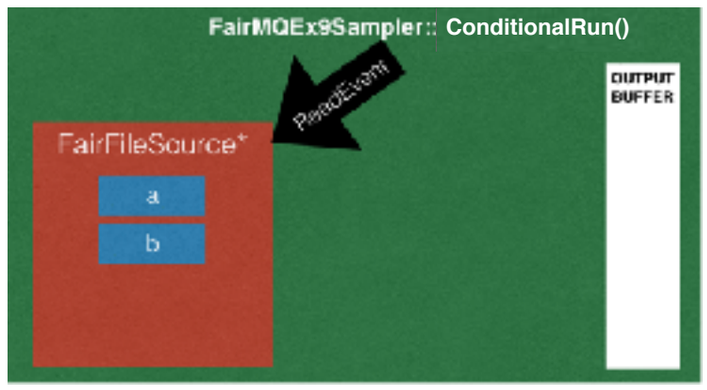
```
fileSink.AddOutputBranch(“FairEventHeader”,”EventHeader.”);
fileSink.AddOutputBranch(“TClonesArray(anyclassname)”,”branchname”);
```

- ParameterMQServer creates instance of the FairRuntimeDb, which reads the parameters from ROOT or ASCII files.

https://github.com/FairRootGroup/FairRoot/tree/dev/examples/MQ/9-PixelDetector

# FairMQEx9Sampler

- InitTask(): creates FairFileSource* fSource with the provided input file names and activates all the needed branches (getting TObject*) in the fSource;

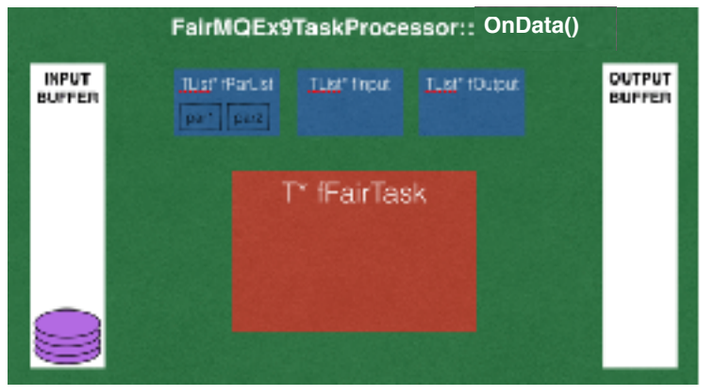



- ConditionalRun(): calls fSource->ReadEvent(): each requested  TObject* is wrapped in a TMessage and added to a multipart FairMQParts message; the result FairMQParts message is sent for each event;

https://github.com/FairRootGroup/FairRoot/tree/dev/examples/MQ/9-PixelDetector

# FairMQEx9TaskProcessor

- InitTask(): creates object fFairTask of class T, asks fFairTask to fill the TList of needed parameters (by calling fFairTask->GetParList()); creates empty input and output TLists;

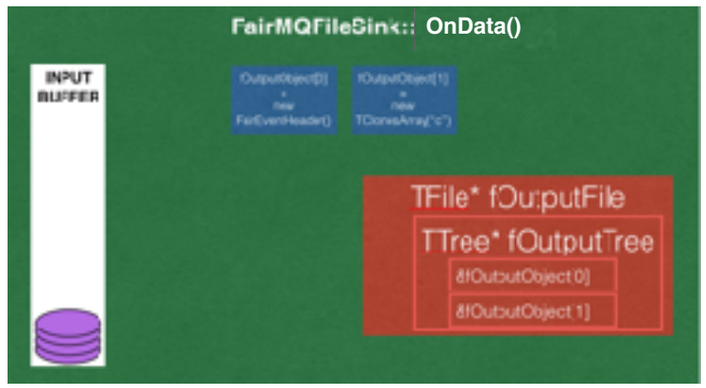



- OnData(): receives the FairMQParts message; runs through its parts to fill the input TList; if FairEventHeader object is found and the RunId changed, the new parameters are requested from the ParameterMQServer and the fFairTask is initialised with the new parameters (by calling fFairTask->InitMQ()); calls the ExecMQ function of the fFairTask; wraps all the TObjects in the output TList in the TMessages and adds them to the output FairMQParts; sends the result output FairMQParts

https://github.com/FairRootGroup/FairRoot/tree/dev/examples/MQ/9-PixelDetector

# FairMQEx9FileSink

- InitTask(): creates output TFile and TTree; adds requested objects to the output TTree;




- OnData(): receives the FairMQParts message; runs through its parts to set the branch addresses to the received objects; fills the output tree;

https://github.com/FairRootGroup/FairRoot/tree/dev/examples/MQ/9-PixelDetector

# Starting MQ devices

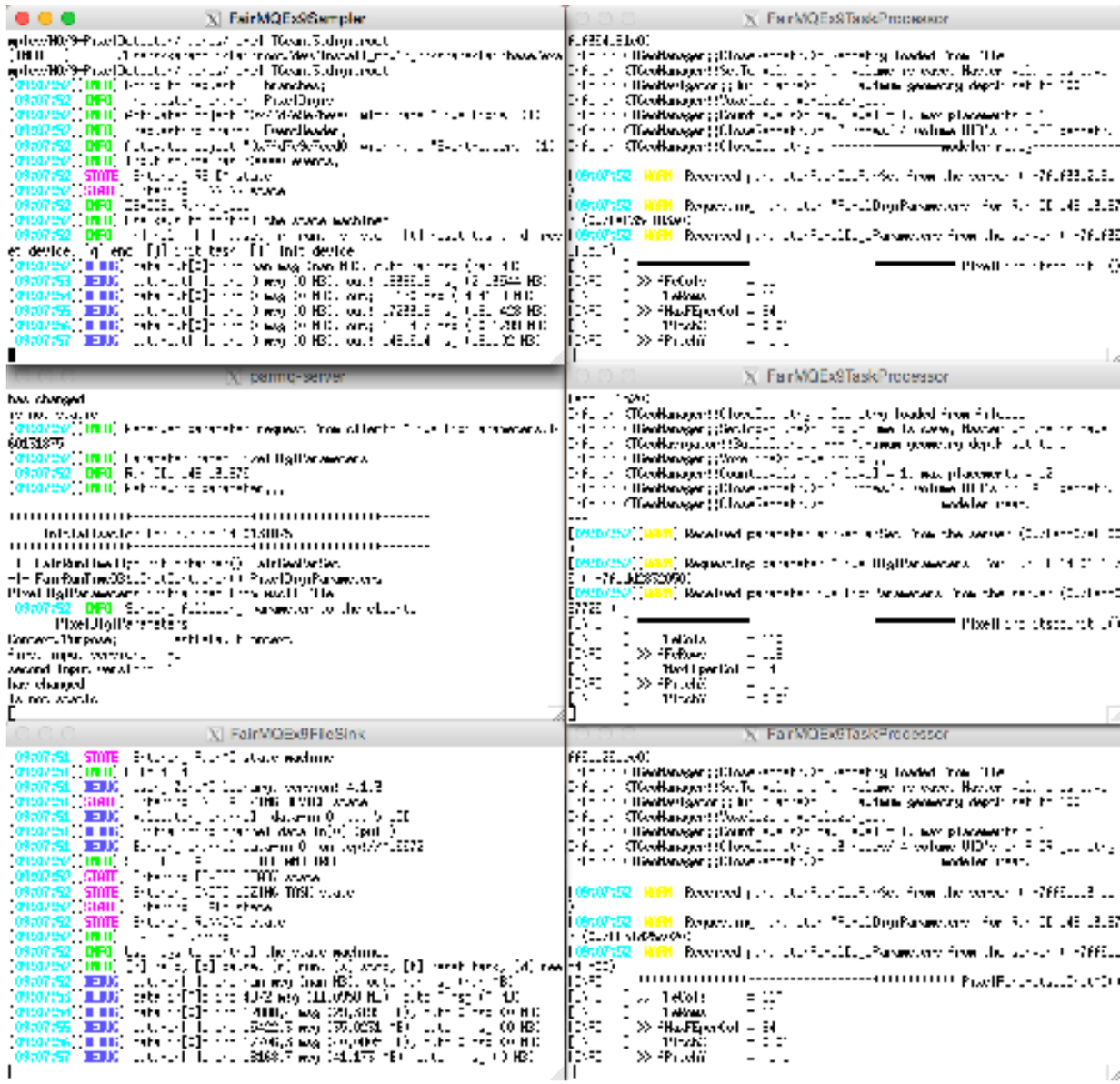
- MQ Devices are run by c++ programs, compiled and stored in the bin folder of your build and install directories.
- Each MQ device has several settings that should be provided at startup:
  - transport protocol (zeromq, nanomsg);
  - verbosity (TRACE, DEBUG*, INFO, NOLOG);
  - unique string ID;
  - connection type (push/pull,req/rep,pub/sub) and method (bind or connect), address and port, buffer sizes (all set in a config file, JSON or XML)
  - plus additional settings of the particular MQ device implementation.
- Example command to start program containing MQ device:

```
./FairMQEx9Sampler --transport zeromq --verbose INFO --id sampler1
--mq-config ../share/fairbase/examples/MQ/9-PixelDetector/run/options/Pixel9MQConfig_Multipart.json
--file-name ../share/fairbase/examples/MQ/9-PixelDetector/macros/pixel_TGeant3.digi.root
--branch-name PixelDigis
```

https://github.com/FairRootGroup/FairRoot/tree/dev/examples/MQ/9-PixelDetector

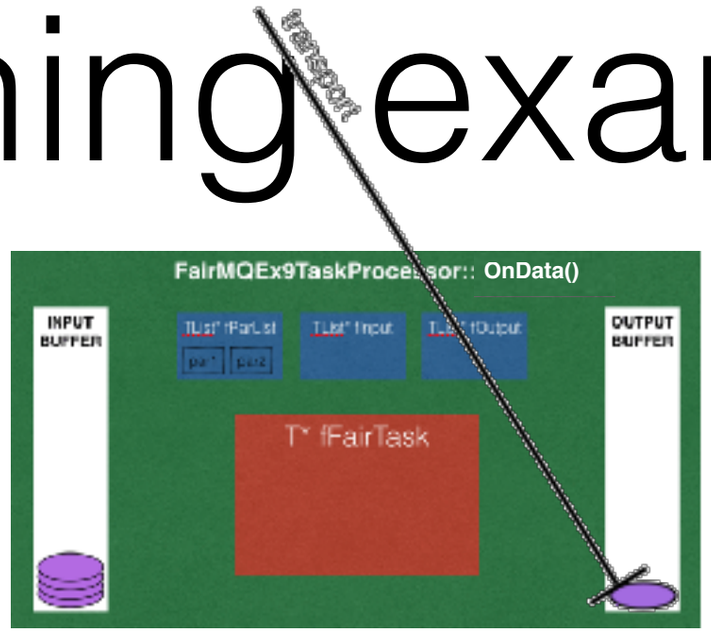
# Shell script

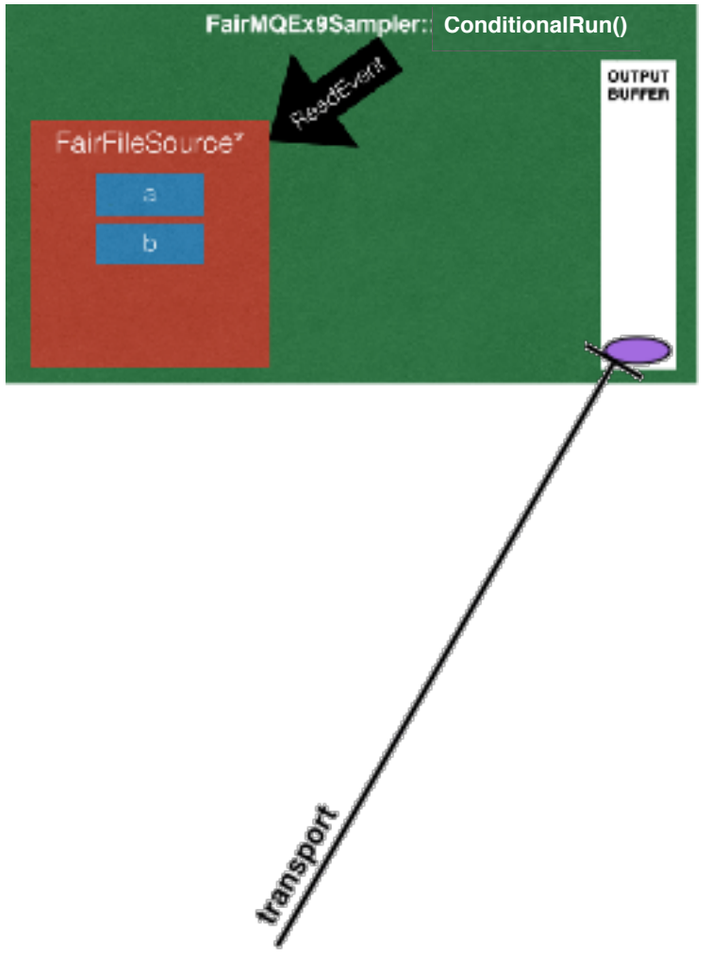
- Specifies program options that are not taken from the JSON file, or that are often changed;
- Starts all the programs from the topology at once;
- Opens several xterm windows and starts one program per window;
- Example execution of the startFairMQEx9.sh script results in:
- Several configurable shell script provided in the 9-PixelDetector/run/scripts/; shell scripts located in: {build,install}/bin/examples/MQ/9-PixelDetector/

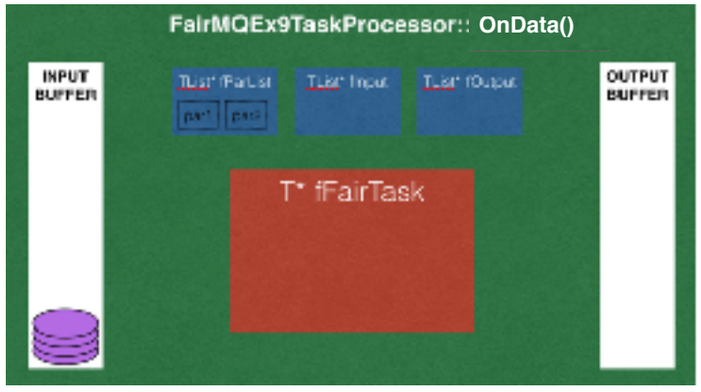


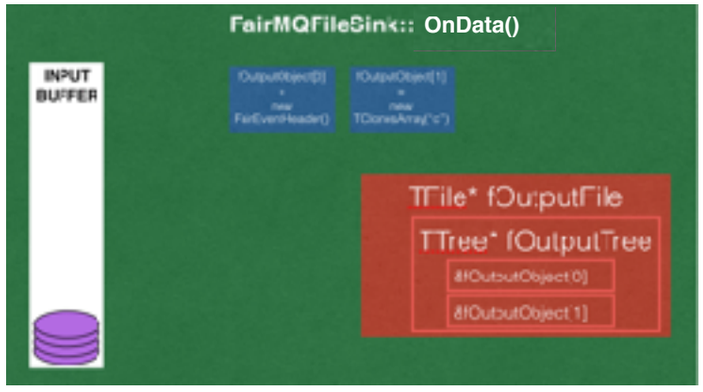
https://github.com/FairRootGroup/FairRoot/tree/dev/examples/MQ/9-PixelDetector

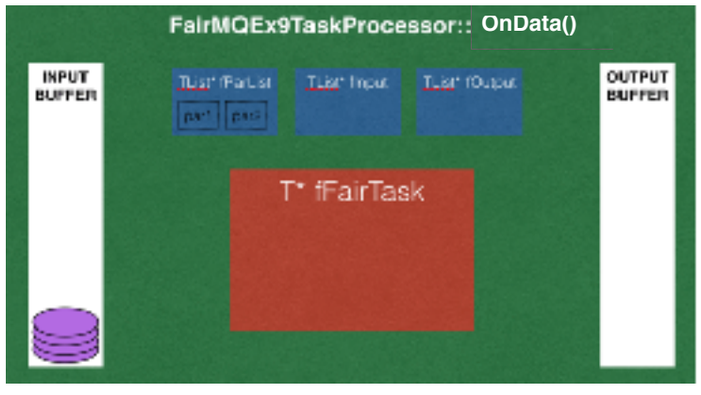
# Running example



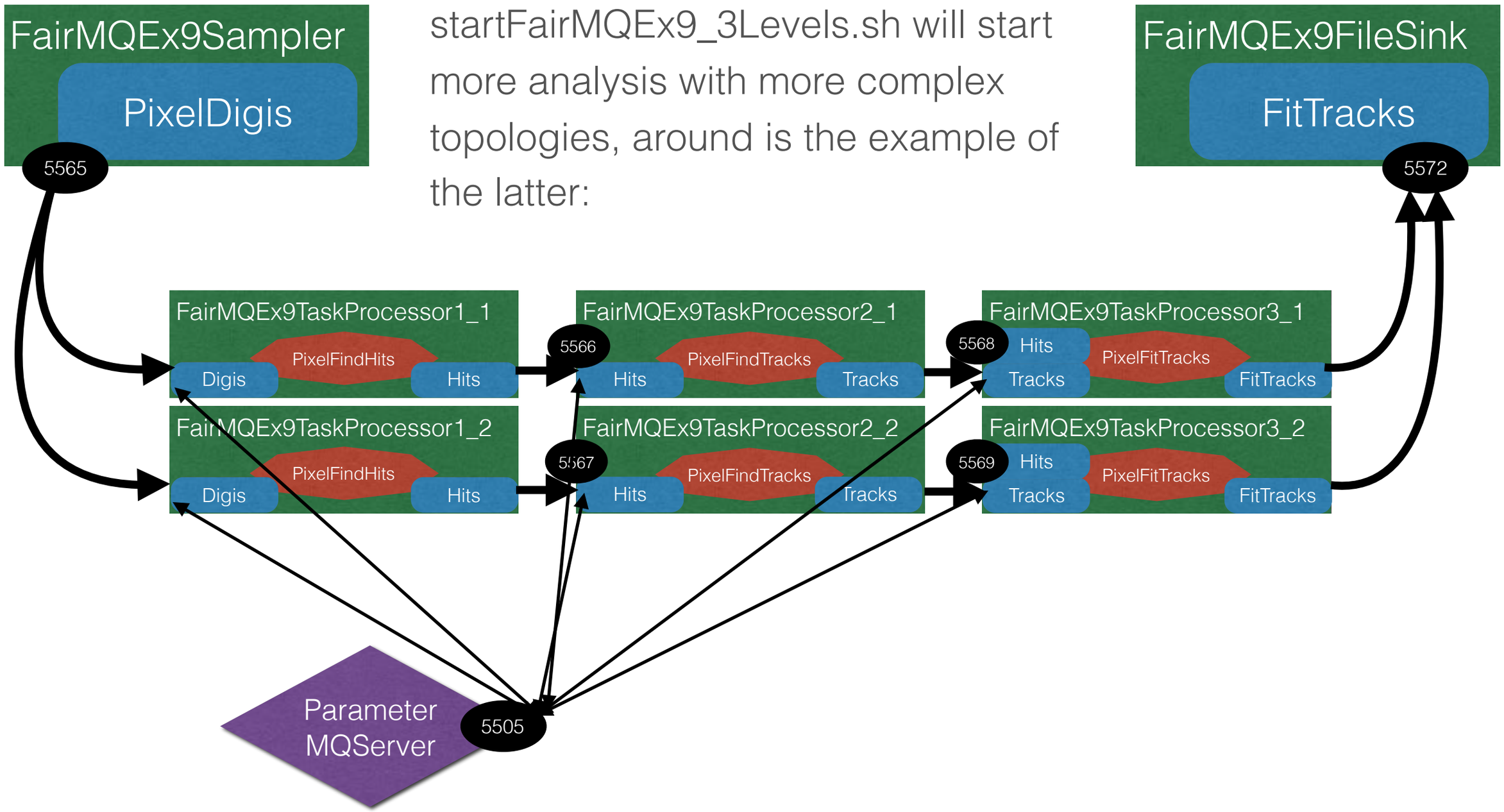



https://github.com/FairRootGroup/FairRoot/tree/dev/examples/MQ/9-PixelDetector

# Other topologies

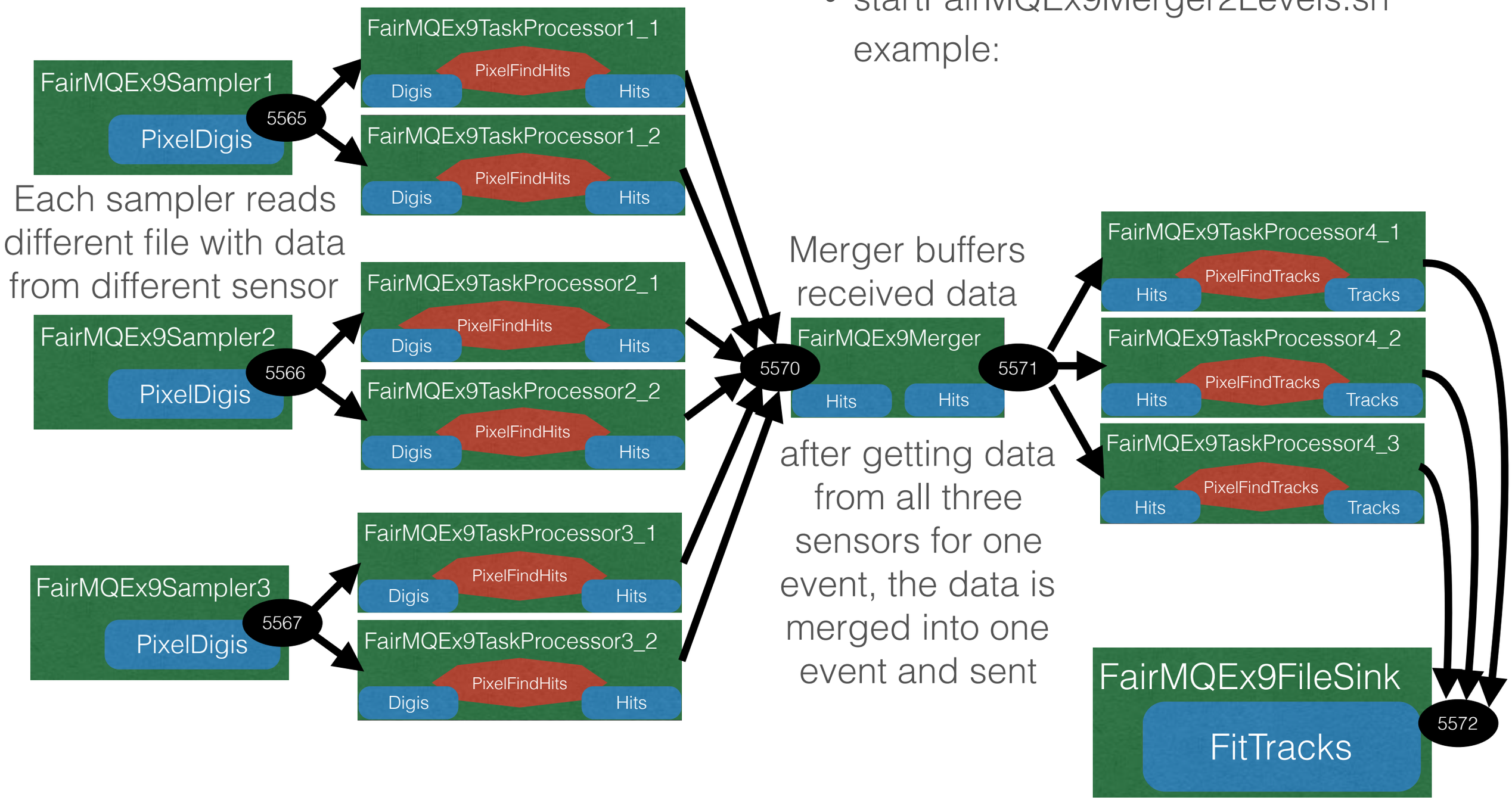
- startFairMQEx9_2Levels.sh or startFairMQEx9_3Levels.sh will start more analysis with more complex topologies, around is the example of the latter:



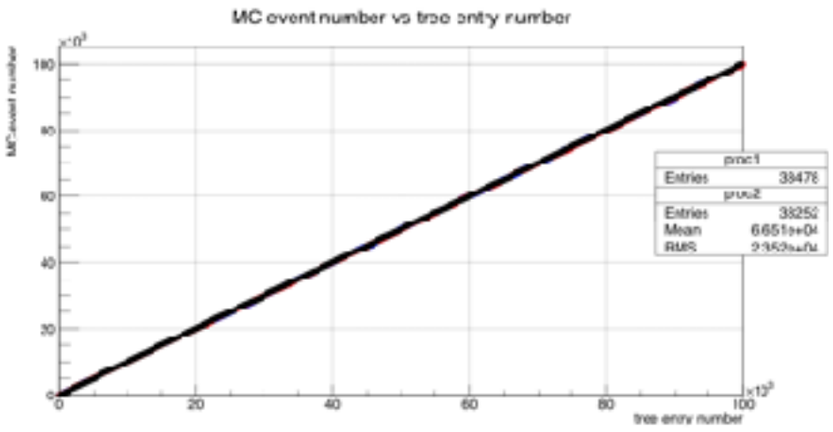
https://github.com/FairRootGroup/FairRoot/tree/dev/examples/MQ/9-PixelDetector

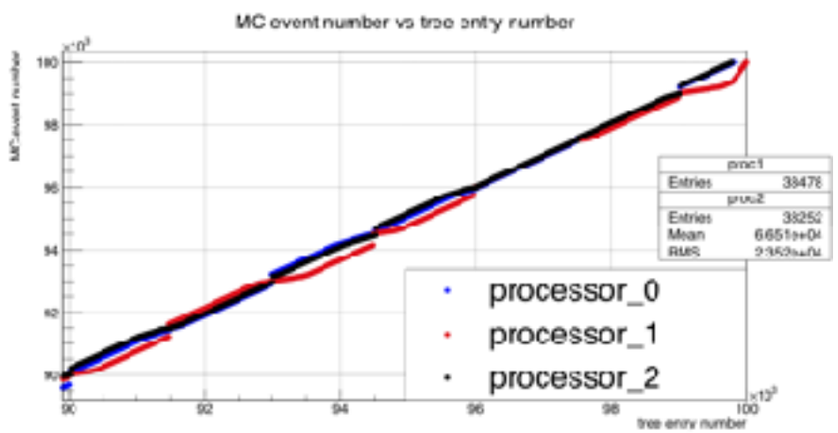
# Other topologies

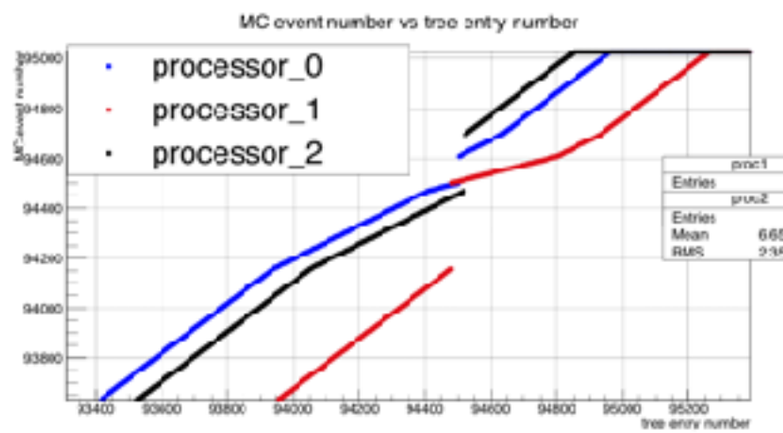
- startFairMQEx9Merger2Levels.sh example:

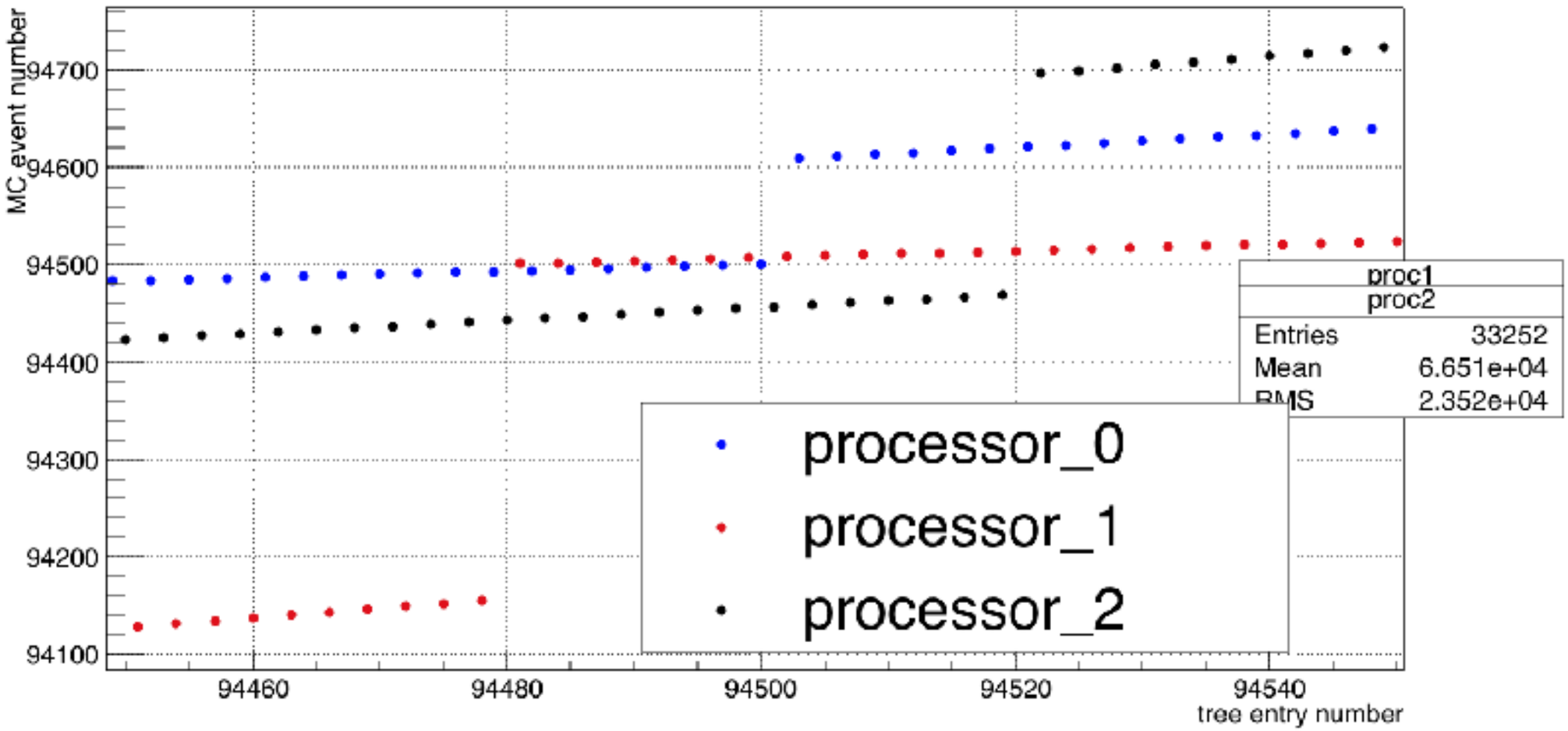


# Event order


MC event number vs tree entry number
MC event number
tree entry number
proc1
Entries 38478
proc2
Entries 33252
Mean 6651e+04

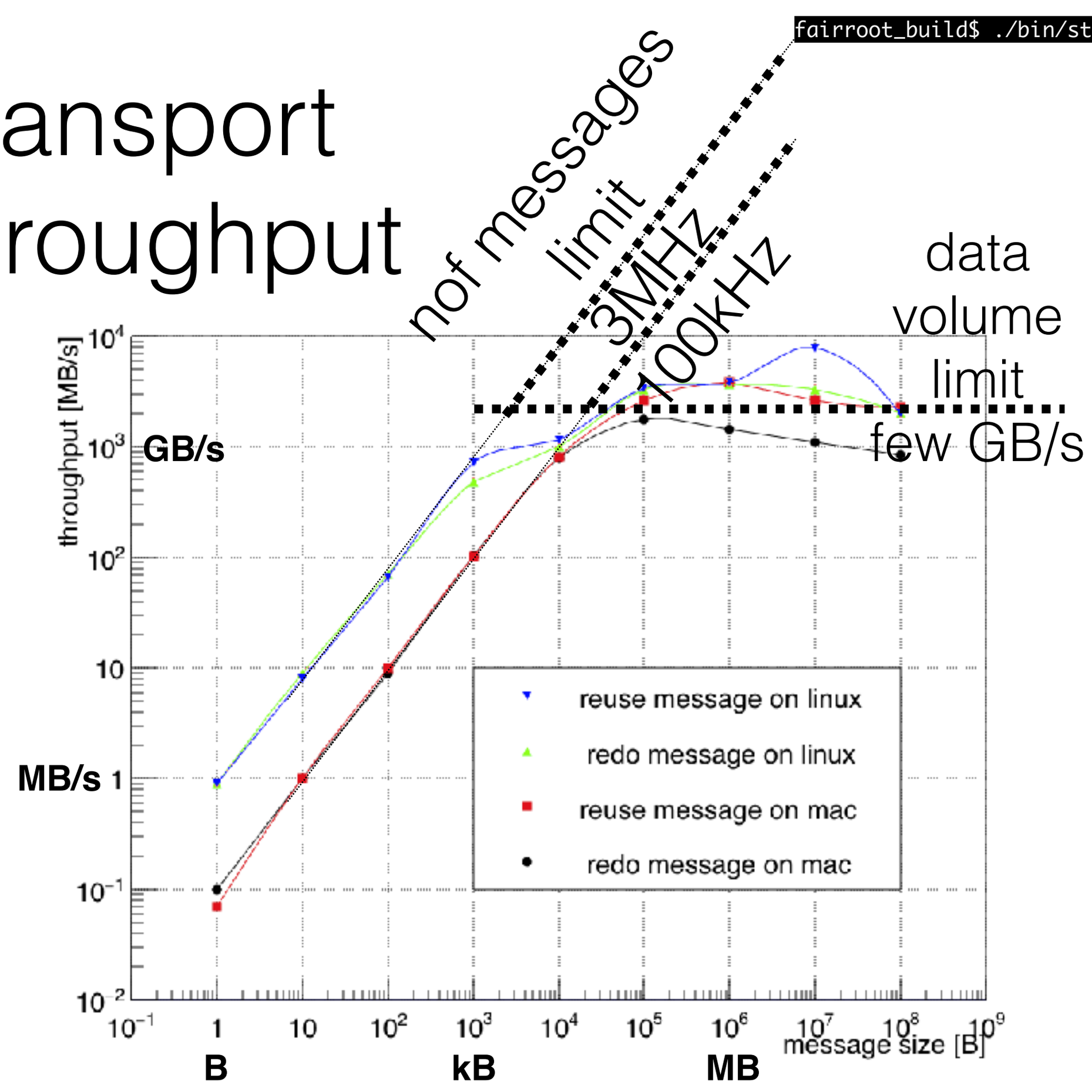

MC event number vs tree entry number
MC event number
tree entry number
proc1
Entries 38478
proc2
Entries 33252
Mean 6651e+04
processor_0
processor_1
processor_2


MC event number vs tree entry number
MC event number
tree entry number
processor_0
processor_1
processor_2
proc1
Entries
proc2
Mean


MC event number vs tree entry number
MC event number
tree entry number
94700
94600
94500
94400
94300
94200
94100
94460
94480
94500
94520
94540
proc1
proc2
Entries 33252
Mean 6.651e+04
RMS 2.352e+04
processor_0
processor_1
processor_2

# Transport throughput

```
fairroot_build$ ./bin/startMQBenchmark.sh
```



https://github.com/FairRootGroup/FairRoot/tree/dev/examples/advanced/Tutorial3

# Serialization

User is free to choose any serialization method for the content of the FairMQMessage.

Four examples are provided to demonstrate serialization with 4 different methods:

| **binary** | **Google protocol buffers** | **boost::serialization (binary)** | **ROOT TMessage** |
|---|---|---|---|
| • Fastest (no serialization)<br>• Requires the data to be in continuous chunks in memory (POD structs/ classes)<br>• Not portable between different architectures | • Fastest serialization<br>• Support for different programming languages<br>• Custom data container has to be precompiled<br>• ~20% slower than binary | • Most flexible<br>• Generic device code<br>• ~30% slower than binary | • Simplest use for TObjects (e.g. TClonesArray)<br>• Generic device code<br>• ~45% slower than binary |

Very rough estimates. Serialization performance depends a lot on the structure of the data!

# DDS

Radoslaw Karabowicz, GSI

http://dds.gsi.de

# **D**ynamic **D**eployment **S**ystem

- implements a single-responsibility-principle command line tool-set and APIs,
- treats users' tasks as black boxes,
- doesn't depend on RMS (provides deployment via SSH, when no RMS is present),
- supports workers behind FireWalls with only outgoing connection,
- doesn't require pre-installation on WNs,
- deploys private facilities on demand with isolated sandboxes,
- provides a key-value properties propagation service for tasks,
- provides a rules based execution of tasks.

# DDS. The contract

The system takes so called “topology file” as the input.

- Users describe desired tasks and their dependencies using topology files,
- users are provided with a WEB GUI to create topos. Can be created manually as well.

http://dds.gsi.de | https://github.com/FairRootGroup/FairRoot/tree/dev/examples/MQ/3-dds

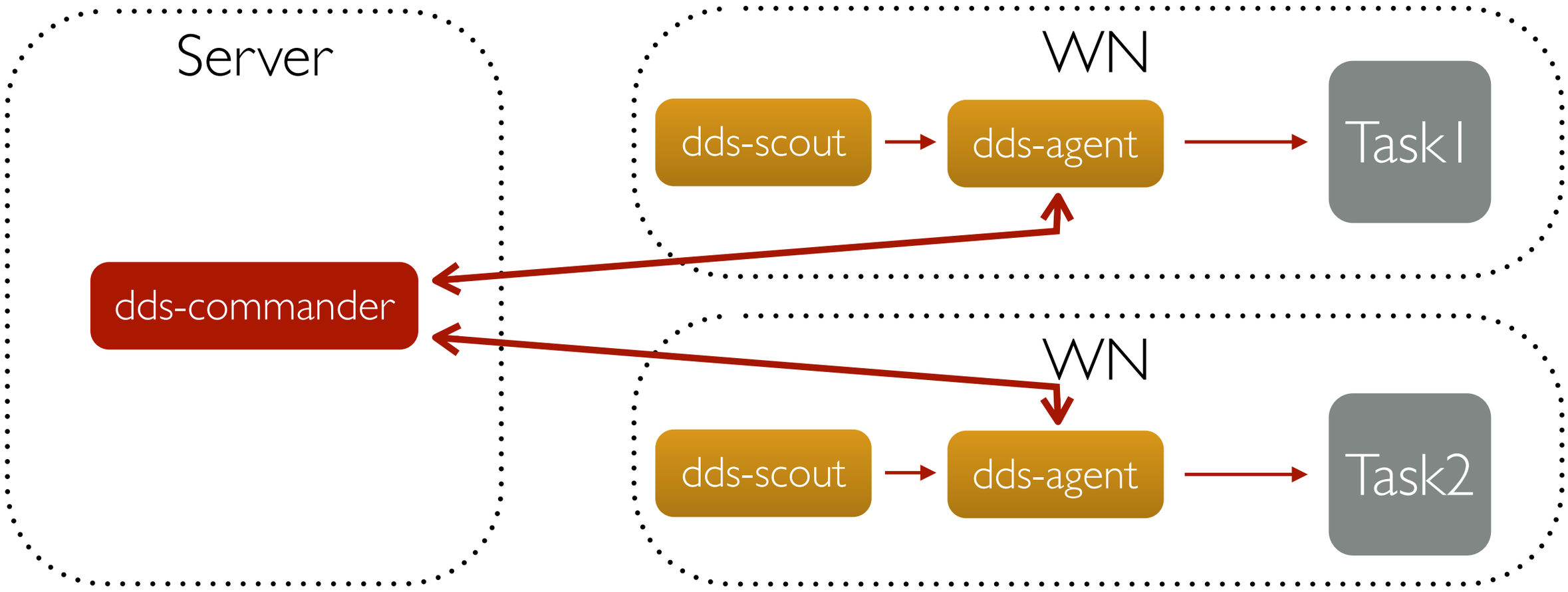
# Running DDS

```
dds-server start -s

dds-submit --rms localhost -n 10

dds-topology --activate topology_file_name.xml

dds-server stop
```

also available: ssh_plugin, work on alternatives in progress:

```
dds-submit --rms ssh --config config_file_name.cfg
```

# DDS. The contract




**dds-server** *start*
**dds-submit** -r *ssh* --ssh-rms-cfg *ssh_hosts.cfg*
**dds-topology** --*activate* *topology_test.xml*
**dds-server** *stop*