

# FairMQ application example

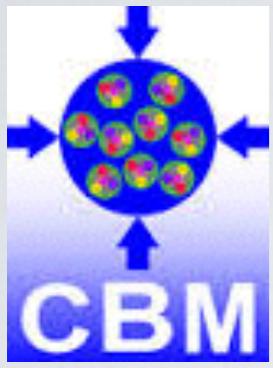
Radoslaw Karabowicz, GSI

# ALFA Framework

- FairMQ is developed as a part of a larger project, the ALFA framework.
- ALFA is developed in a collaboration between the FairRoot Group, experiments at FAIR and ALICE.
- The data-flow based model is grounded on the Message Queues based multi-processing (FairMQ).
- Provides configuration, process management and monitoring tools.
- Provides unified access to configuration parameters and databases.



ALICE

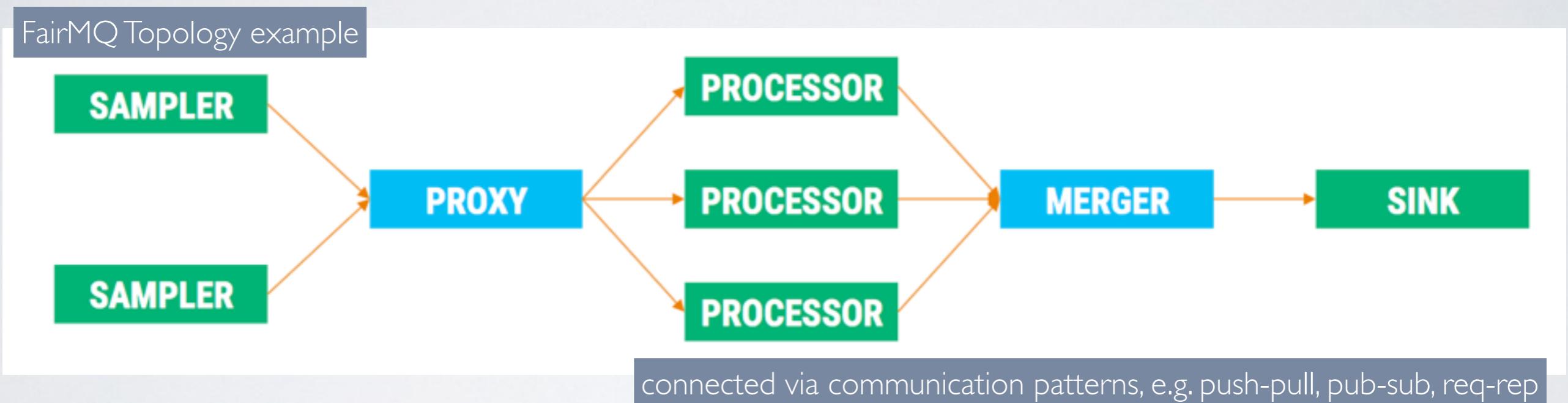


R<sup>3</sup>B



# What is FairMQ?

- A data transport framework to organize processing **tasks** in **topologies**, consisting of independent processes (**Devices**), communication via asynchronous message queues over network or inter-process channels.



- Ready to use devices are available for common scenarios.
- User-defined devices can be implemented by inheriting from **FairMQDevice** class.

# What is FairMQ?

- Defines workflow for the process.
- Promotes separation of concerns to increase maintainability and reduce coupling: simple, single responsibility devices.
- Keeps the user and transport code separate, allowing them to evolve independently: abstract transport interface, user task separation.
- Simplifies and unifies handling of the transport details: socket settings, polling, termination, access to communication patterns.
- Takes care of the device state (change, query), termination, command interface, device configuration, logging.

# motivation behind MQ/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;

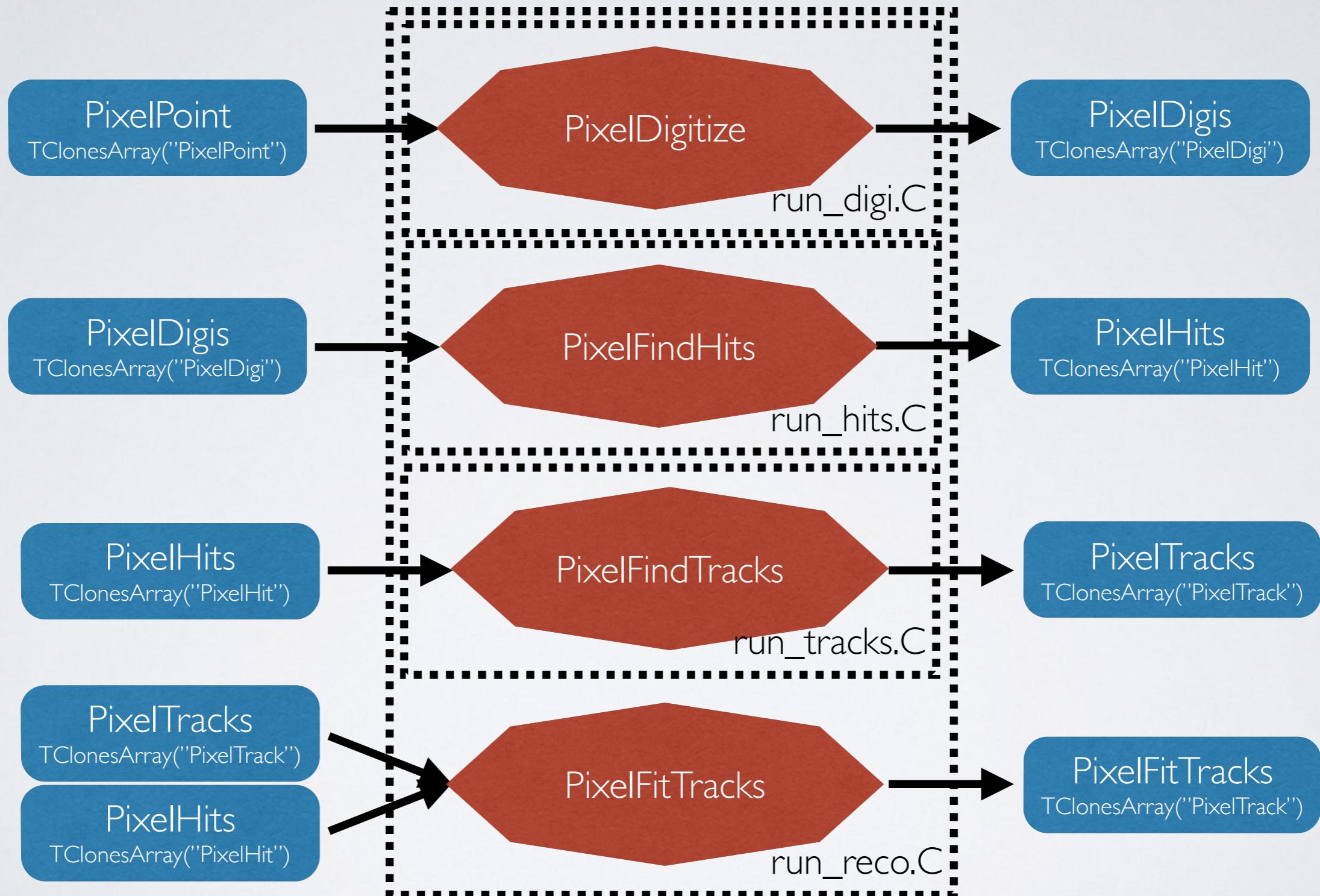
# /examples/MQ/9-PixelDetector

- 3 stations with 4 rectangular sensor each:
  - size:  $5 \times 5 \text{cm}^2$ , inner hole:  $1 \times 1 \text{cm}^2$ ,  
at  $z = 5\text{cm}$ ;
  - size:  $10 \times 10 \text{cm}^2$ , inner hole:  $1 \times 1 \text{cm}^2$ ,  
at  $z = 10\text{cm}$ ;
  - size:  $20 \times 20 \text{cm}^2$ , inner hole:  $2 \times 2 \text{cm}^2$ ,  
at  $z = 20\text{cm}$ ;
- each sensor divided into pixels ( $0.01 \times 0.01 \text{cm}^2$ ), that are grouped into FE modules (110 pixels  $\times$  116 pixels)

FEs numbering on 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 ] Digits:      940281  ( 9.40281 per event )
[INFO ]
[INFO ] ----- Pixel Hit Finder : Summary -----
[INFO ] Events:      100000
[INFO ] Digits:      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 ] -----
```

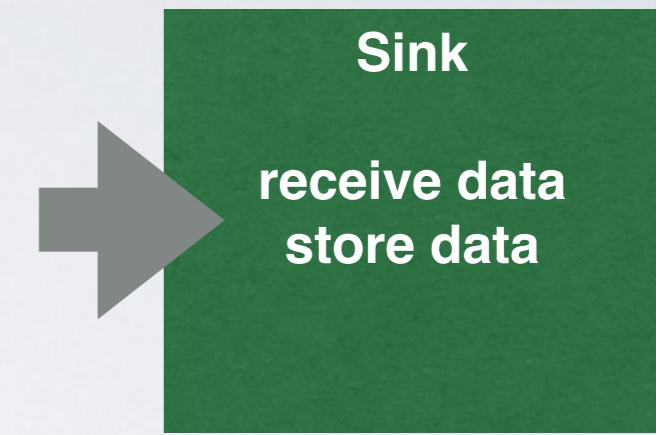
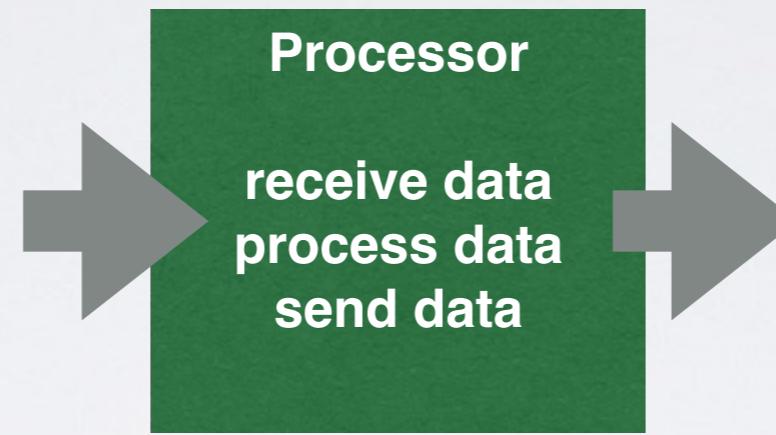
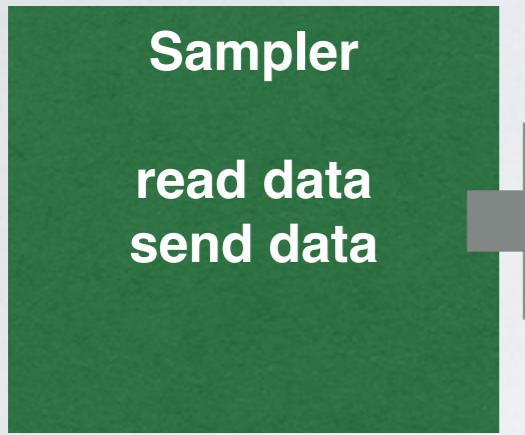
# data classes, tasks and macros



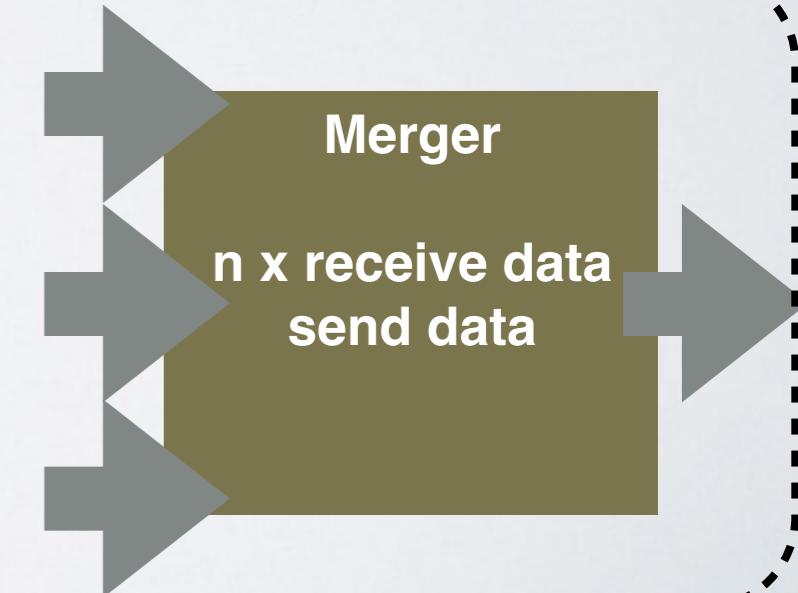
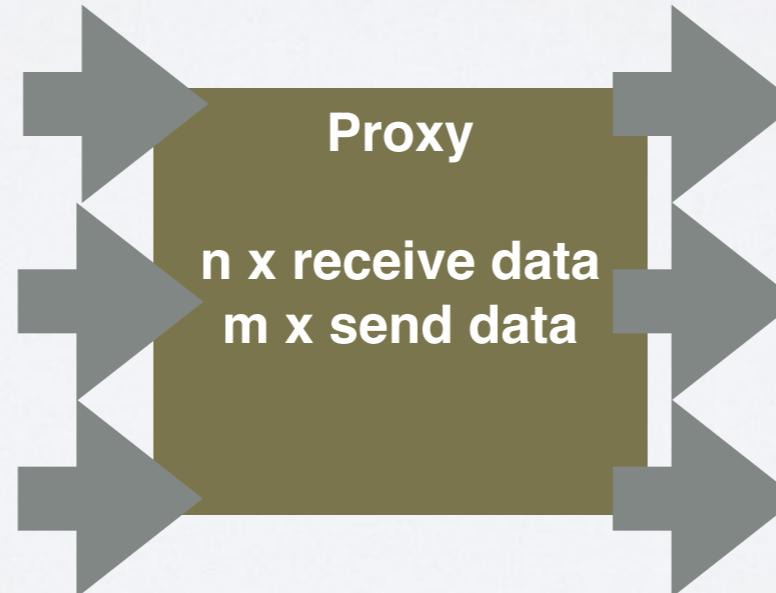
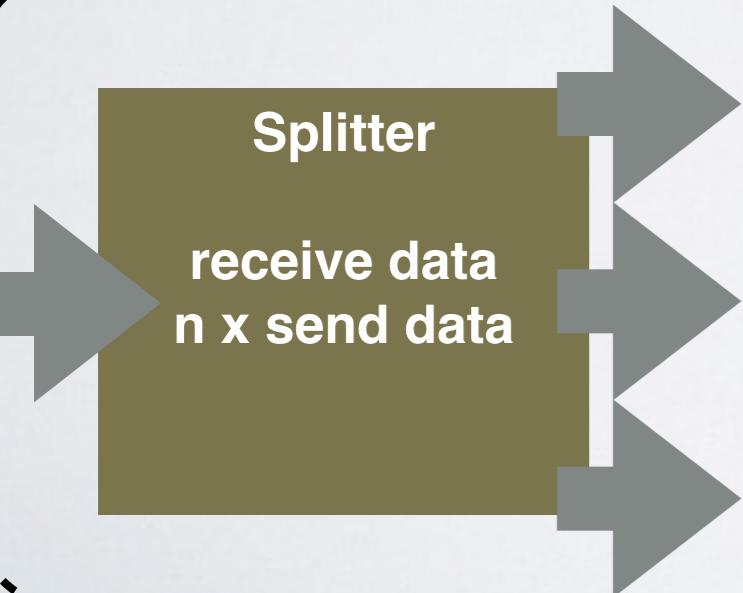
# MQ Devices

## General types

**touching the data**



**not touching the data**



# MQ concepts

- **FairMQParts** - multi-part message that is sent between devices;
- **TMessage** - a ROOT wrapper for any TObject that can be added to FairMQParts (another options: binary, protobuf, boost);

# MQ Devices for FairRoot 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.

# FairMQEx9Sampler

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

# template<typename T> 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`;
- `Run()`: 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`

# FairMQEx9FileSink

- `InitTask()`: creates output `TFile` and `TTree`; adds requested objects to the output `TTree`;
- `Run()`: receives the `FairMQParts` message; runs through its parts to set the branch addresses to the received objects; fills the output tree;

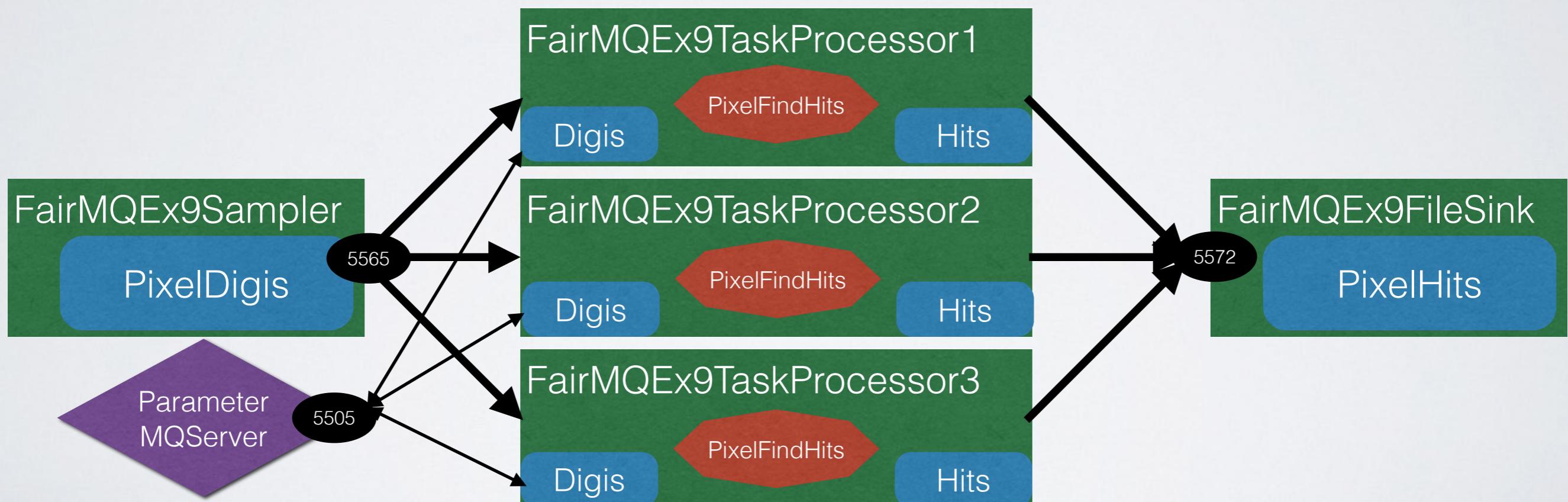
# Starting MQ devices

- 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
```

# Topology

- Defined in the JSON or XML file (one may write parser of any config file);
- Contains list of devices that we need in order to analyse data;
- Specifies unique IDs of devices, connection types, addresses and ports, buffer sizes;
- Example view of the topology with 1 samplers, 3 processors, 1 files sink and parameter server:



- Several topologies provided in the 9-PixelDetector/run/options.

# 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 startFairMQEx9New.sh script results in:
- Several configurable shell script provided in the 9-PixelDetector/run/scripts; shell scripts located in build/bin and install/bin

```
FairMQEx9Sampler
[INFO] - /Users/karabow/fairroot/dev/install_nov15_r5/share/Fairbase/examples/HG/9-PixelDetector/macros/pixel_TGeant3.digi.root
[INFO] Going to request 2 branches:
[INFO] requesting branch "PixelDigits"
[INFO] Activated object "0x7fd7e9e7be00" with name "PixelDigits" (1)
[INFO] requesting branch "EventHeader."
[INFO] Activated object "0x7fd7e9e7ced0" with name "EventHeader." (1)
[INFO] Input source has 500000 events.
[STATE] Entering READY state
[STATE] Entering RUNNING state
[INFO] DEVICE: Running...
[INFO] Use keys to control the state machine:
[INFO] [h] help, [p] pause, [r] run, [s] stop, [t] reset task, [d] reset device, [q] end, [j] init task, [i] init device
[DEBUG] data-out[0]: in: nan msg (nan MB), out: nan msg (nan MB)
[DEBUG] data-out[0]: in: 0 msg (0 MB), out: 18365.6 msg (20.3544 MB)
[DEBUG] data-out[0]: in: 0 msg (0 MB), out: 22150 msg (24.4139 MB)
[DEBUG] data-out[0]: in: 0 msg (0 MB), out: 17233.8 msg (19.0428 MB)
[DEBUG] data-out[0]: in: 0 msg (0 MB), out: 18334.7 msg (20.1799 MB)
[DEBUG] data-out[0]: in: 0 msg (0 MB), out: 14615.4 msg (16.102 MB)

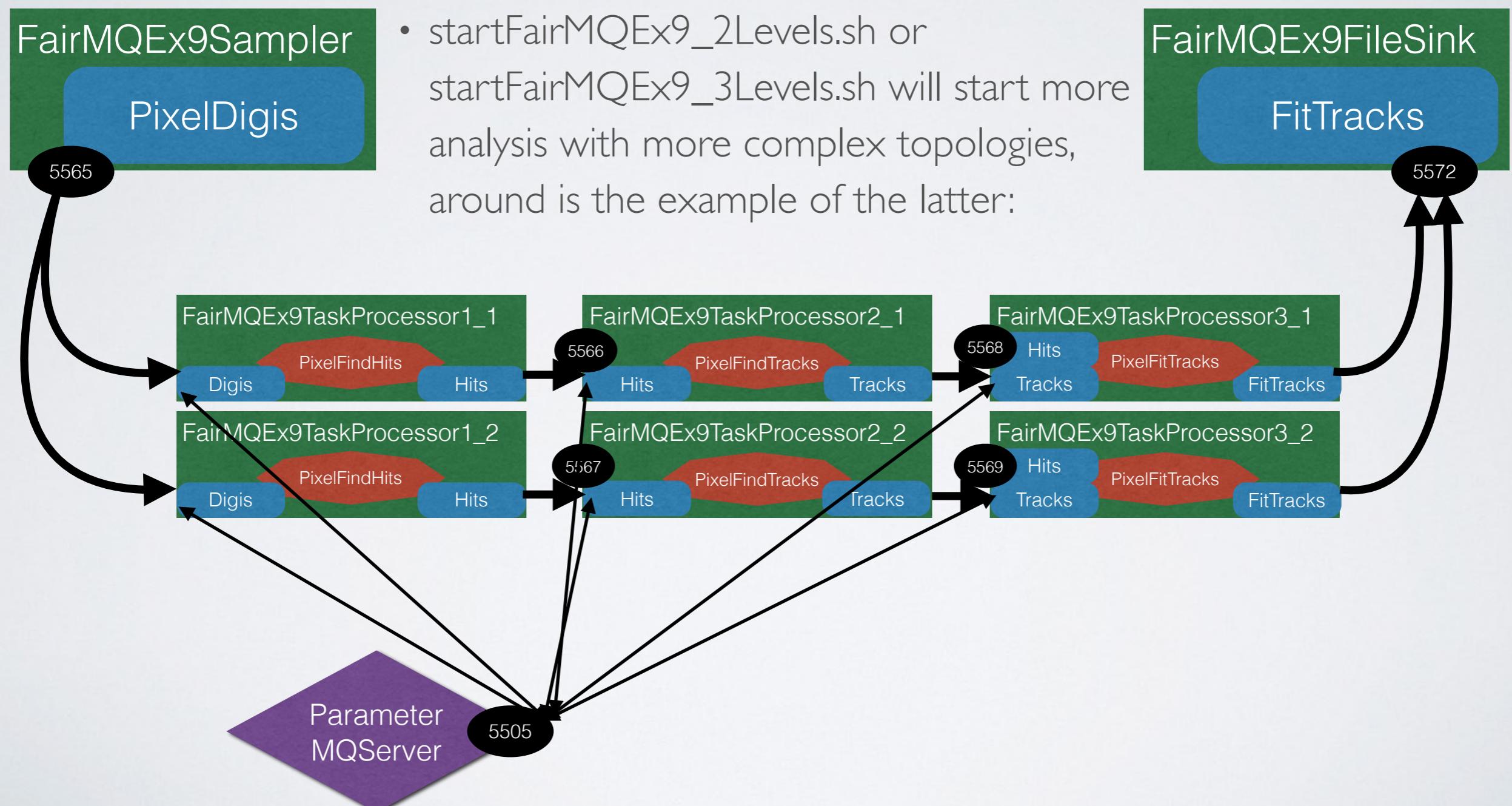
FairMQEx9TaskProcessor
fef354181c0)
Info in <TGeoManager::CloseGeometry>; Geometry loaded from file...
Info in <TGeoManager::SetTopVolume>; Top volume is cave, Master volume is cave
Info in <TGeoNavigator::BuildCache>; --- Maximum geometry depth set to 100
Info in <TGeoManager::Voxelize>; Voxelizing...
Info in <TGeoManager::CountLevels>; max level = 1, max placements = 12
Info in <TGeoManager::CloseGeometry>; 13 nodes/ 4 volume UID's in FAIR geometry
Info in <TGeoManager::CloseGeometry>; -----modeler ready-----
[09:07:52][WARN] Received parameterFairGeoParSet from the server (0x7fef33e219c0)
[09:07:52][WARN] Requesting parameter "PixelDigiParameters" for Run ID 1460131875 (0x7fef354183e0)
[09:07:52][WARN] Received parameterPixelDigiParameters from the server (0x7fef354183e0)
[INFO] **** PixelFindHits::InitIO()
[INFO] >> FFeCols = 110
[INFO] >> FFeRows = 116
[INFO] >> FMaxFEperCol = 64
[INFO] >> FPitchX = 0.01
[INFO] >> FPitchY = 0.01

FairMQEx9TaskProcessor
fed2831b20)
Info in <TGeoManager::CloseGeometry>; Geometry loaded from file...
Info in <TGeoManager::SetTopVolume>; Top volume is cave, Master volume is cave
Info in <TGeoNavigator::BuildCache>; --- Maximum geometry depth set to 100
Info in <TGeoManager::Voxelize>; Voxelizing...
Info in <TGeoManager::CountLevels>; max level = 1, max placements = 12
Info in <TGeoManager::CloseGeometry>; 13 nodes/ 4 volume UID's in FAIR geometry
Info in <TGeoManager::CloseGeometry>; -----modeler ready-----
[09:07:52][WARN] Received parameterFairGeoParSet from the server (0x7fed07ef300)
[09:07:52][WARN] Requesting parameter "PixelDigiParameters" for Run ID 1460131875 (0x7fed2832050)
[09:07:52][WARN] Received parameterPixelDigiParameters from the server (0x7fed07ef300)
[INFO] **** PixelFindHits::InitIO()
[INFO] >> FFeCols = 110
[INFO] >> FFeRows = 116
[INFO] >> FMaxFEperCol = 64
[INFO] >> FPitchX = 0.01
[INFO] >> FPitchY = 0.01

FairMQEx9FileSink
ff51d25bee0)
Info in <TGeoManager::CloseGeometry>; Geometry loaded from file...
Info in <TGeoManager::SetTopVolume>; Top volume is cave, Master volume is cave
Info in <TGeoNavigator::BuildCache>; --- Maximum geometry depth set to 100
Info in <TGeoManager::Voxelize>; Voxelizing...
Info in <TGeoManager::CountLevels>; max level = 1, max placements = 12
Info in <TGeoManager::CloseGeometry>; 13 nodes/ 4 volume UID's in FAIR geometry
Info in <TGeoManager::CloseGeometry>; -----modeler ready-----
[09:07:52][WARN] Received parameterFairGeoParSet from the server (0x7ff51b2e30dc0)
[09:07:52][WARN] Requesting parameter "PixelDigiParameters" for Run ID 1460131875 (0x7ff51d25c020)
[09:07:52][WARN] Received parameterPixelDigiParameters from the server (0x7ff51b25c020)
[INFO] **** PixelFindHits::InitIO()
[INFO] >> FFeCols = 110
[INFO] >> FFeRows = 116
[INFO] >> FMaxFEperCol = 64
[INFO] >> FPitchX = 0.01
[INFO] >> FPitchY = 0.01
```

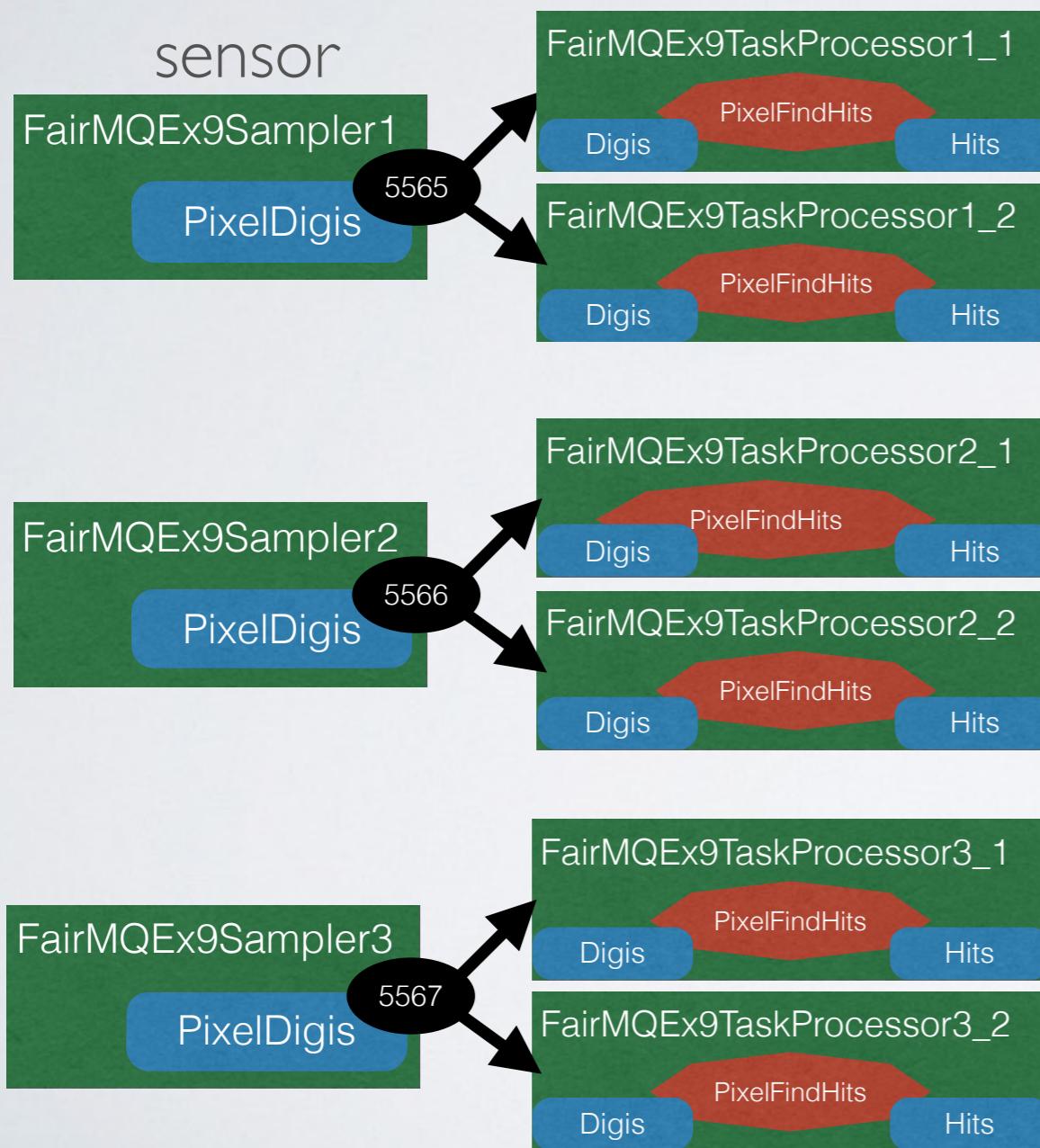
- startFairMQEx9New.sh - shell script to start sampler, few processors, sink and parameter server. The task to process may be easily changed by using --task-name PixelFindTracks or --task-name PixelFitTracks;

## Other topologies:



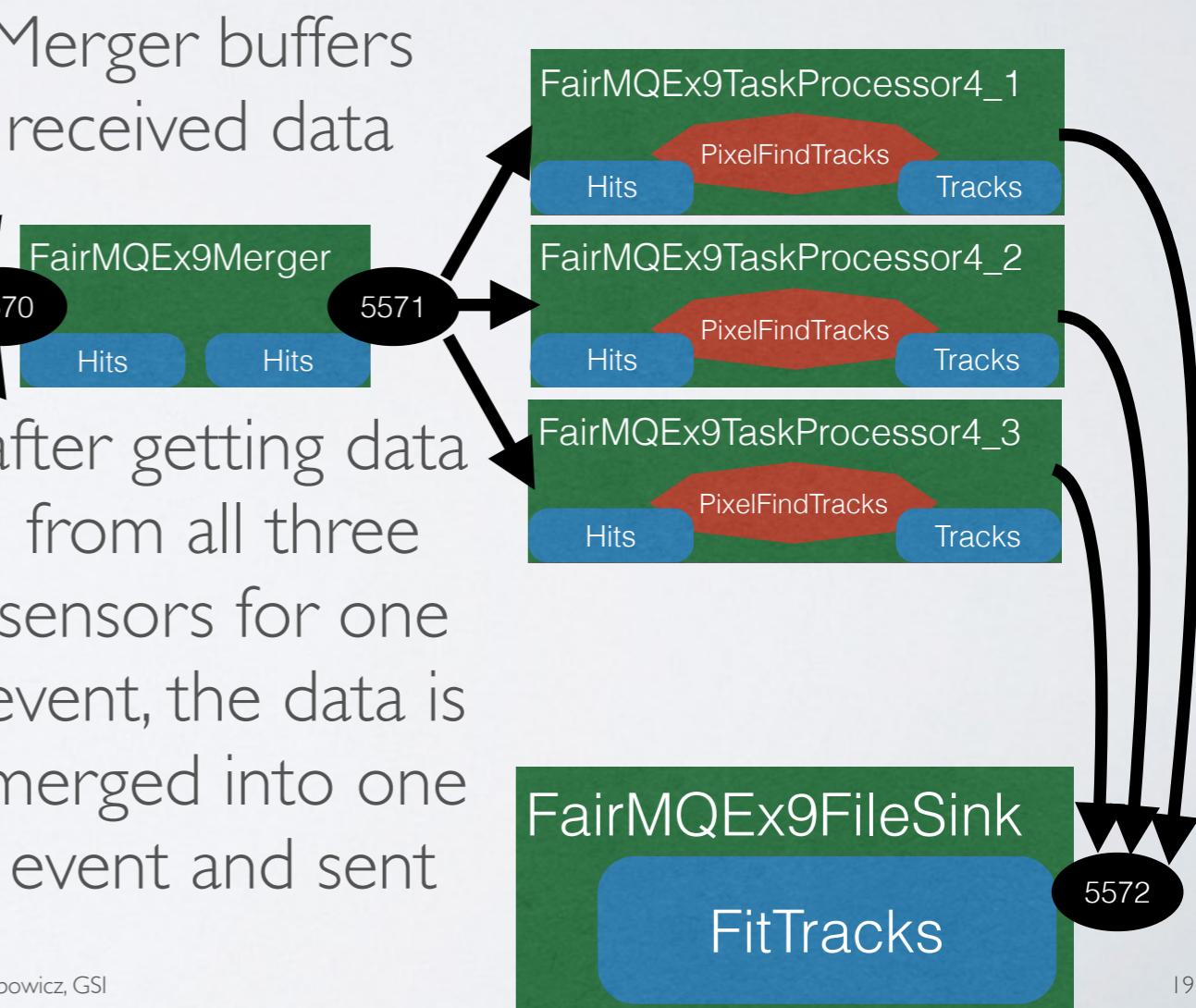
# Other topologies:

Each sampler reads from different file with data from one sensor



Merger buffers received data

after getting data from all three sensors for one event, the data is merged into one event and sent



# How to start?

- Example available in the FairRoot dev branch:

<https://github.com/FairRootGroup/FairRoot/tree/dev>

- Simulate and digitize data in  
examples/MQ/9-PixelDetector/macros/

- Run provided examples, f.e.:

```
<< nov15p4_r5 - - fairroot/pixel9 >> - bash - 100x13
karabowi@kp3mac004:~/fairroot/pixel9/build_nov15p4_r5_DDS$ ./bin/startFairMQEx9_Static.sh

|   server |   sampler |   processor1 |   processor2 |   processor3 |   file sink |
| pid =  5953 | pid =  5954 | pid =  5955 | pid =  5956 | pid =  5957 | pid =  5958 |
| %CPU      mem |
| 0.0  53692 | 81.5 157864 | 52.5 140688 | 51.2 141560 | 52.5 143364 | 139.9 201192 |

Jobs finished in 10 seconds.
Finished gracefully.
There are 100000 in the input file.
There are 100000 in the output file.
Shell script finished successfully.
karabowi@kp3mac004:~/fairroot/pixel9/build_nov15p4_r5_DDS$
```

run devices in background, monitor running, check output

```
<< nov15p4_r5 - - fairroot/pixel9 >> - sleep - 98x39
-
Thank you for using this script (for 5291 seconds).
karabowi@kp3mac004:~/fairroot/pixel9/build_nov15p4_r5_DDS$ ./bin/controlDDS.sh
RUNNING->s->READY->t->DEVICE_READY->d->IDLE->q /x-Exit
[ norma-server ] [ processor_0 ] [ processor_1 ] [ processor_2 ] [ sampler ] [ sink ]
[          ] [          ] [          ] [          ] [          ] [          ]
<< nov15p4_r5 - - fairroot/pixel9 >> - bash - 98x39
karabowi@kp3mac004:~/fairroot/pixel9/build_nov15p4_r5_DDS$ dds-server start -s
Checking availability of WN bin of the local system...
found compatible WN bin: dds-wrk-bin-1.1.55.g101a9ce-Darwin-universal.tar.gz
Starting DDS commander...
-----
DDS commander server: 4924

karabowi@kp3mac004:~/fairroot/pixel9/build_nov15p4_r5_DDS$ dds-submit --rms localhost -n 6
dds-submit: Contacting DDS commander on kp3mac004.gsi.de:20003 ...
dds-submit: Connection established.
dds-submit: Requesting server to process job submission...
dds-submit: Server reports: Creating new worker package...
dds-submit: Server reports: RMS plug-in: /Users/karabowi/dds/1.1.42.g482cf62/plugins/dds-submit-localhost/dds-submit-localhost
dds-submit: Server reports: Initializing RMS plug-in...
dds-submit: Server reports: RMS plug-in is online. Startup time: 33ms.
dds-submit: Server reports: Plug-in: Will use the local host to deploy 6 agents
dds-submit: Server reports: Plug-in: Using '/var/folders/nf/xl_sxlm17xl26qqcgs2qb2cr0000gn/T/dds_2016-06-13-30-23-341' to spawn agents
dds-submit: Server reports: Plug-in: Starting DDSScout in '/var/folders/nf/xl_sxlm17xl26qqcgs2qb2cr0000gn/T/dds_2016-06-13-30-23-341/wn'
dds-submit: Server reports: Plug-in: DDS agents have been submitted
dds-submit: Server reports: Plug-in: Checking status of agents...
dds-submit: Server reports: Plug-in: All agents have been started successfully
karabowi@kp3mac004:~/fairroot/pixel9/build_nov15p4_r5_DDS$ dds-topology --set bin/ex9-dds-topology.xml
dds-topology: Contacting DDS commander on kp3mac004.gsi.de:20003 ...
dds-topology: Connection established.
dds-topology: Requesting server to set a new topology...
karabowi@kp3mac004:~/fairroot/pixel9/build_nov15p4_r5_DDS$ dds-topology --activate
dds-topology: Contacting DDS commander on kp3mac004.gsi.de:20003 ...
dds-topology: Connection established.
dds-topology: Requesting server to activate user tasks...
[=====] 100 % (6/6)
Activated tasks: 6
Errors: 0
Total: 6
Time to Activate: 0.013 s
karabowi@kp3mac004:~/fairroot/pixel9/build_nov15p4_r5_DDS$
```

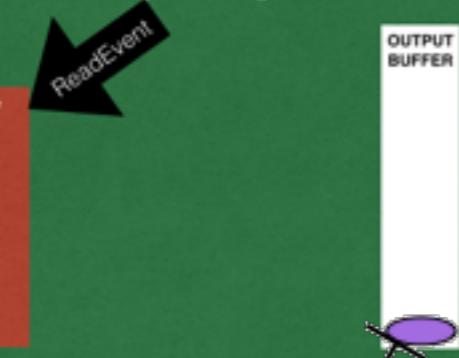
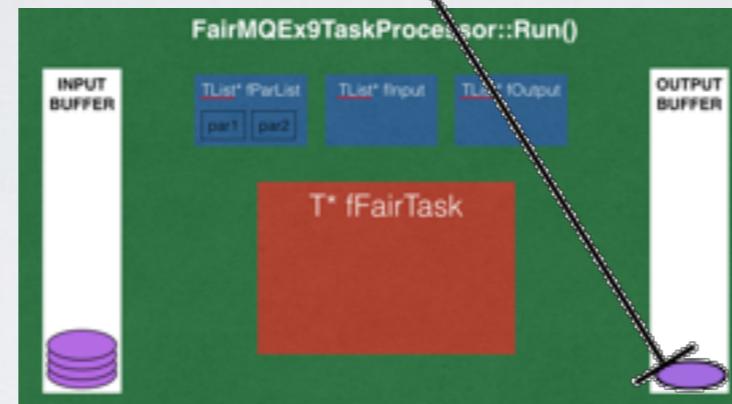
run and control devices with DDS

# Summary

- Ready to use example implemented;
- The example devices are quite simple and yet general enough to allow processing of different tasks;
- Merger of splitted event data implemented;
- Binary transport available for selected data classes;
- The example may be run in DDS:

<https://github.com/FairRootGroup/DDS>

# Thank you for attention



*transport*

Special thanks to the **FairRoot** group:  
 Mohammad Al-Turany  
 Alexey Rybalchenko  
 Nicolas Winckler  
 Anar Manafov  
 Andrey Lebedev  
 Florian Uhlig  
 Dmytro Kresan  
 Thorsten Kollegger

