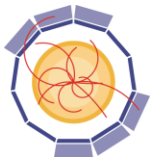
Institute of High Energy Physics
Chinese Academy of Sciences



AIDA 2020

AIDA-2020 First annual meeting

Status and future of EUDAQ

Yi Liu yi.liu@desy.de

On behalf of EUDAQ developers

Code repository https://eudaq.github.io

EUDAQ Feature

- A common data acquisition framework
- OS independent: Linux, Mac OSX, Windows
- Integratable with DUT DAQ systems of independent of their technology
- Modular and flexible design
- “Run Control” via GUI, also CLI interface available
- Online DQM using the “OnlineMonitor”
- Hardware communication done by “producers”
- Used by many groups

EUDAQ ~~Feature~~ Future

- A common data acquisition framework
  - Not only used in EUDET telescope
- OS independent: Linux, Mac OSX, Windows
  - CPU Architecture independent x86 & ARM
- Integration of DAQ systems of DUT independent of the DUT's technology
- Modular and flexible design
  - Processor framework, data processing code is encapsulated as a Processor
- DAQ Run Control via GUI, also CLI interface available
- Online DQM using the OnlineMonitor
  - interface for DQM4HEP standalone online monitor / LCIO support
- Hardware communication done by "producers"
  - Data collected by multiple "DataCollector" instead of singleton
- Used by many groups
  - Flexible "Event Builder"

EUDAQ on EUDET telescope

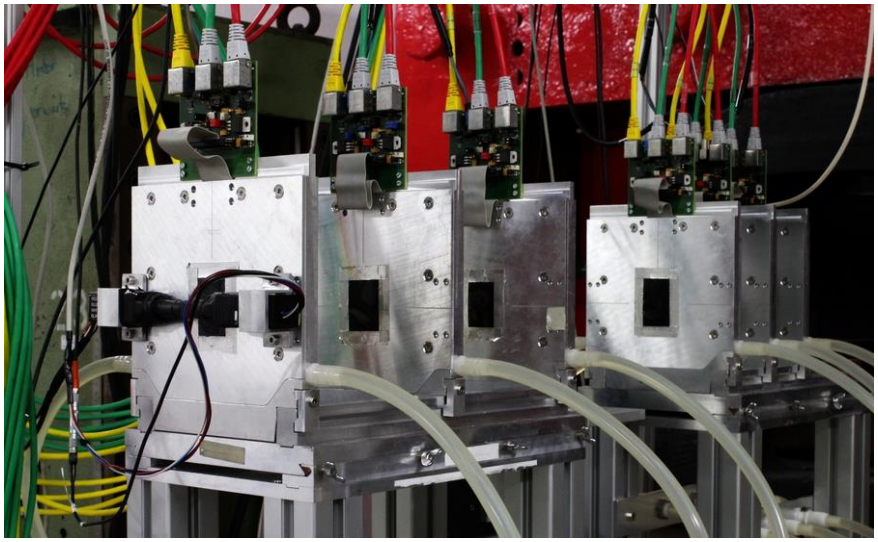
Typical using:

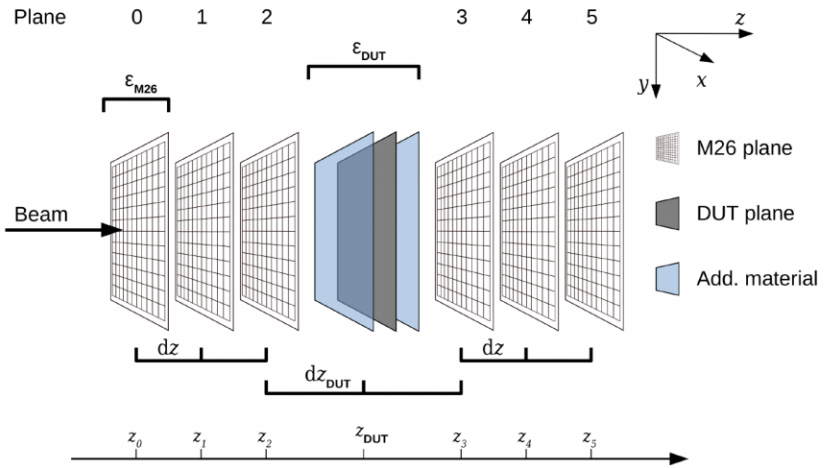
EUDAQ is originally developed as a DAQ system for EUDET-type telescope.

A system trigger signal is distributed in whole EUDET.

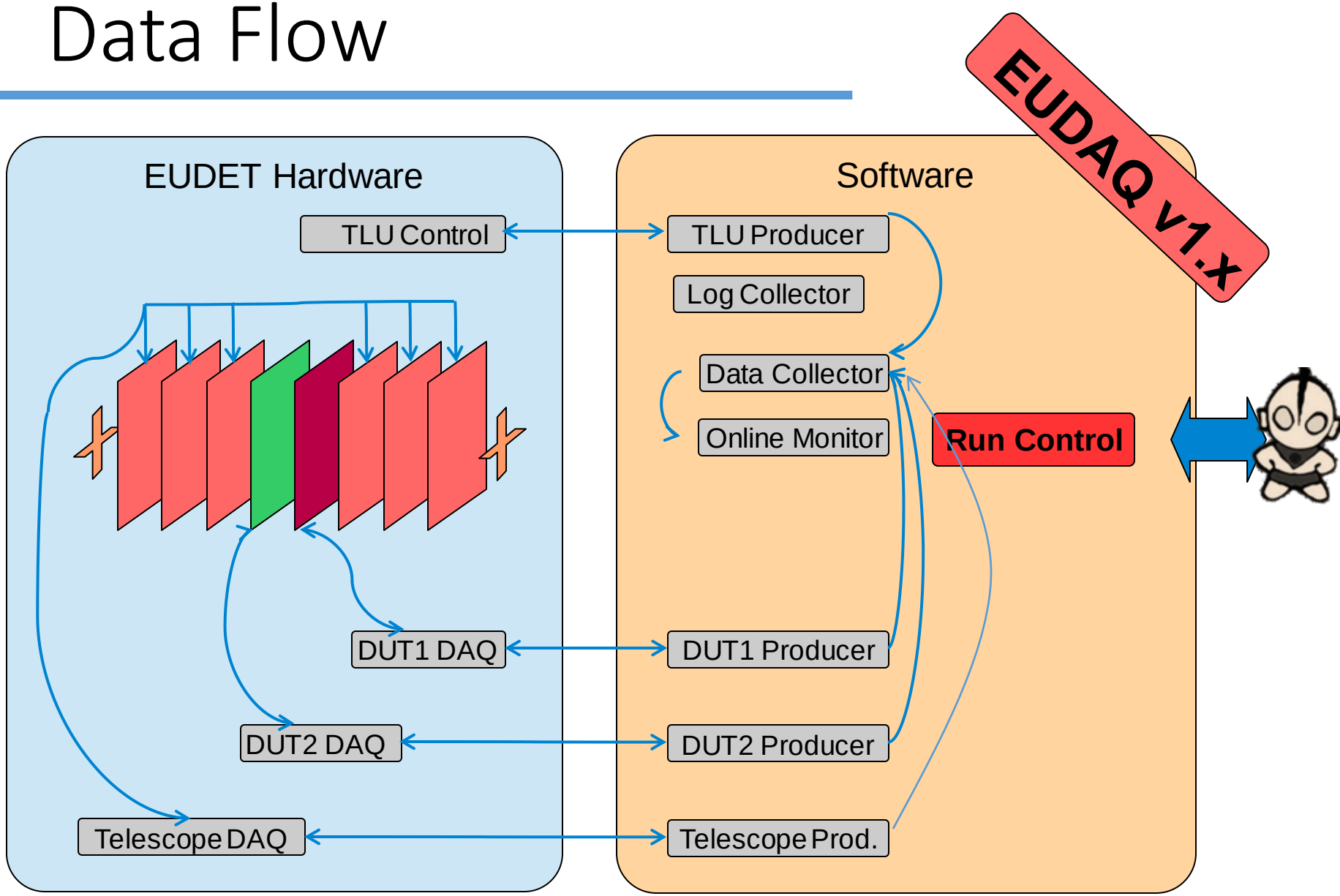
EUDAQ use trigger-ID to merge data and then writes raw-data to hard disk.

EUTelescope analyzes the data from EUDAQ and provide particle track.

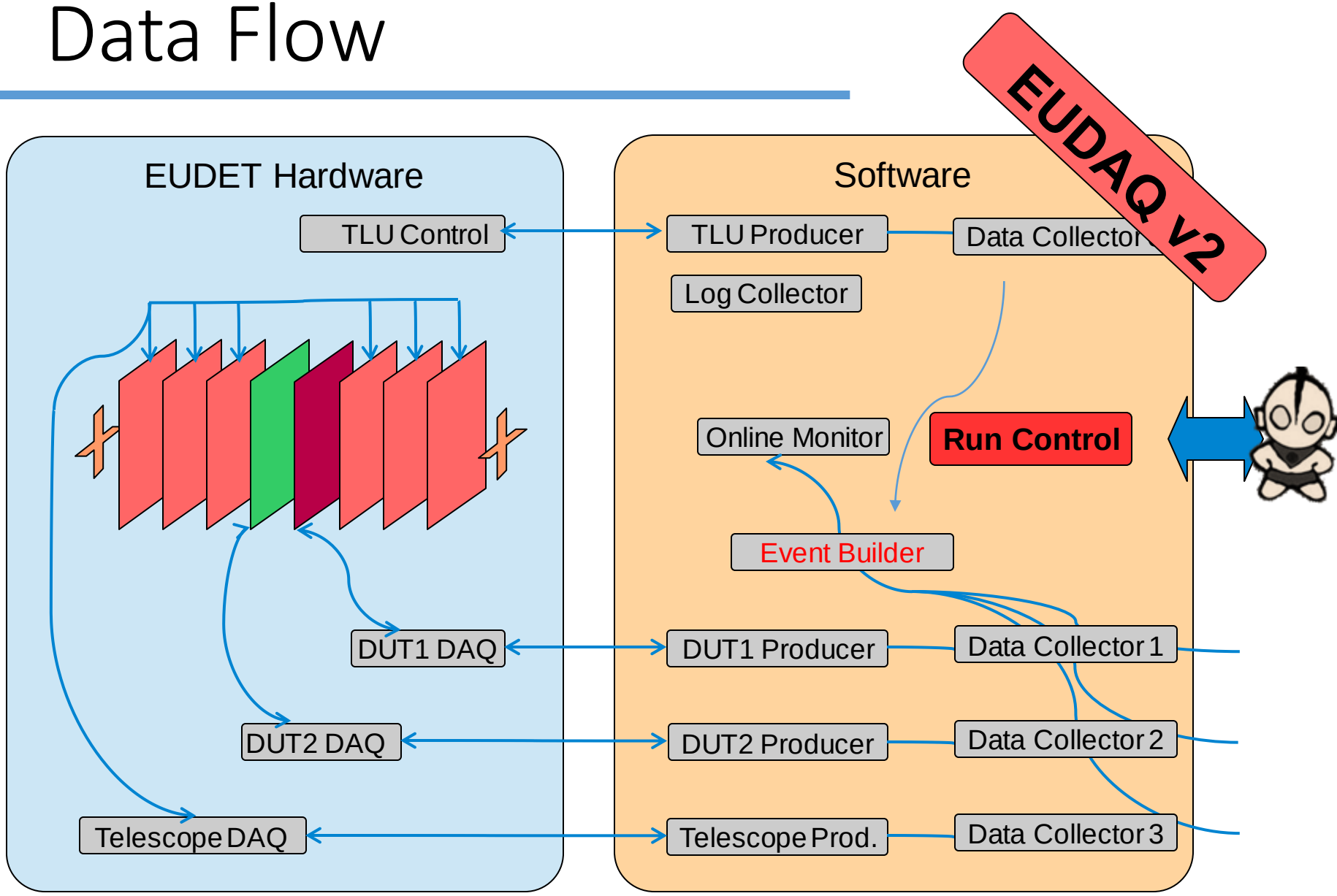




Data Flow


EUDAQ v1.x
EUDET Hardware
Software
TLU Control
TLU Producer
Log Collector
Data Collector
Online Monitor
Run Control
DUT1 DAQ
DUT1 Producer
DUT2 DAQ
DUT2 Producer
Telescope DAQ
Telescope Prod.

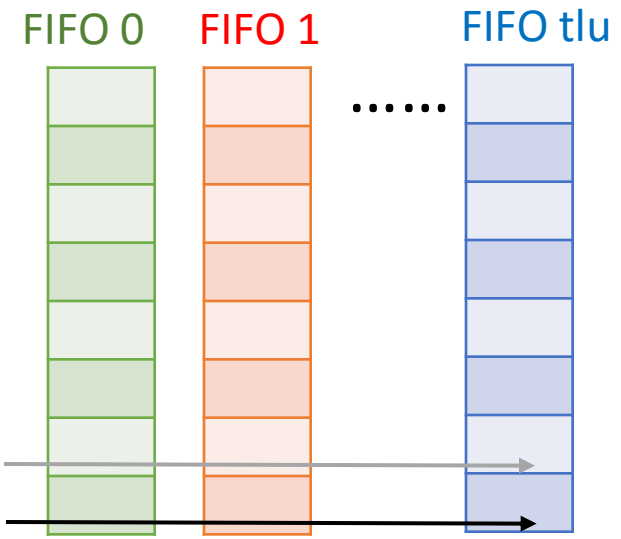
Data Flow


EUDAQ v2
EUDET Hardware
Software
TLU Control
TLU Producer
Data Collector
Log Collector
Online Monitor
Run Control
Event Builder
DUT1 DAQ
DUT1 Producer
Data Collector 1
DUT2 DAQ
DUT2 Producer
Data Collector 2
Telescope DAQ
Telescope Prod.
Data Collector 3

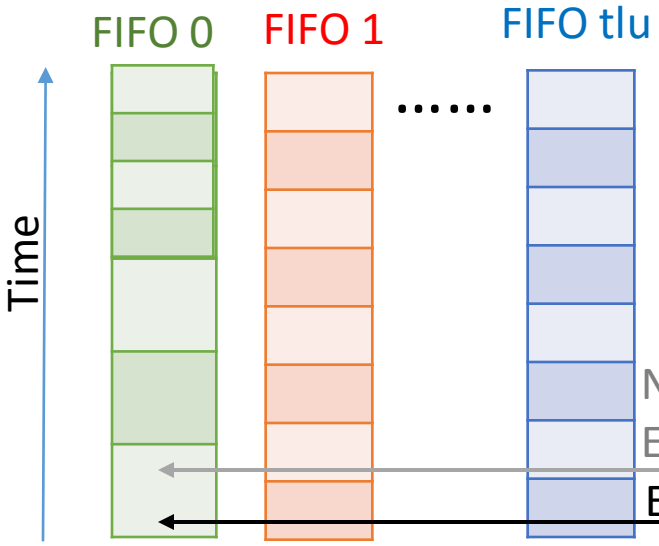
Event Builder

EUDAQv1 case:

- Same event rate from sub-detector is mandated.
- One system trigger ensures the same event rate.
- DataCollector just merges them by one by one.



EUDAQv2 case:



- No unified system trigger
- Different data rate from sub-detectors
- Self triggered sub-detector may exist (FIFO 0)

Event Builder

Merge data from sub-detectors to one according to (Trigger-ID, Timestamp……..)

- The proposed (standard/simple) event builder only supports a standard user case
  Sub-detector provides a least trigger-id or timestamp in their data.

Device	Example Device	Data info
TLU	miniTLU	trigger ID, timestamp
Device triggered by TLU	EUDET telescope	hit data, trigger ID
Self triggered Device	FEI-4	hit data, timestamp

- If not the standard/simple case:
  User should be responsible for the data building logic
  EUDAQv2 provides the framework

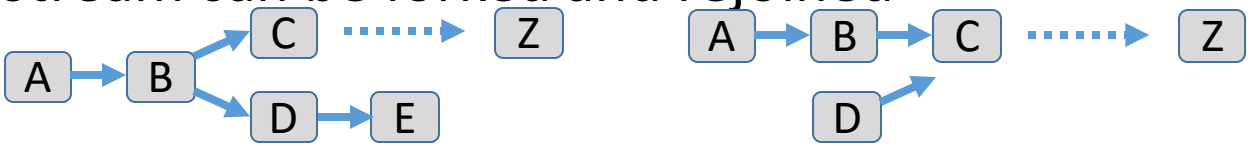
Discussion with AHCAL community makes us aware that there is completely different event building logic and online-monitor requirement, beside EUDET style.

Processor framework (the solution for "Event Builder")

Data processing code is encapsulated as a Processor

Processors are linked to a chain to processing data stream.

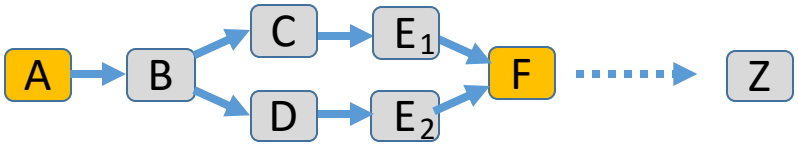
Data stream can be forked and rejoined



Run Independently

"I do everything myself."

Multiple threads capability and safety.



Automatic thread instances:
A: root processor
F: join processor

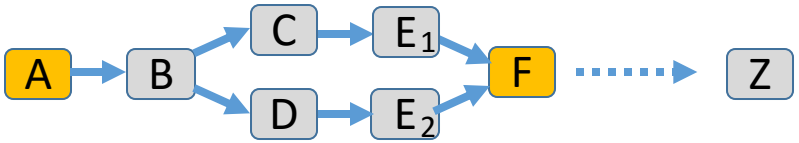
User is completely unware of the threads when the derived processor class is implemented. Threading or NOT, is depending on the location where the processor is.

Processor thread

Data processing code is encapsulated as a Processor

3 threads may be instanced in a processor.

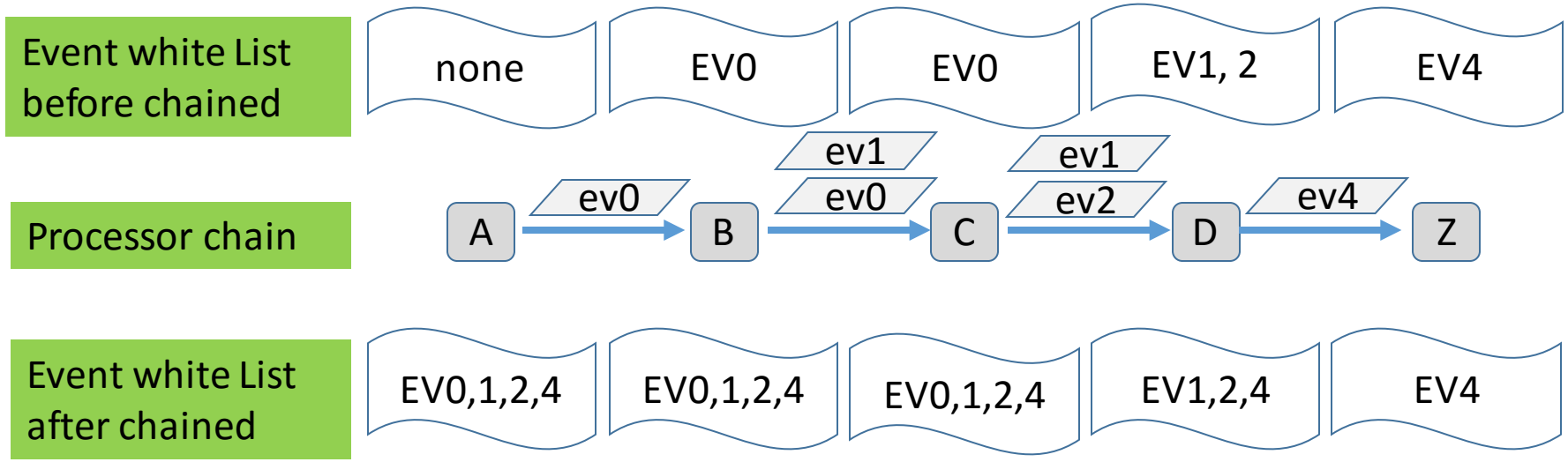
- Producer thread: produce Event from Network, File or Hardware.
- Customer thread: optional enable for a heavy load data processing
- Hub thread: atomically instanced if processor is placed as root or rejoin node



Processor location	Mandate thread	Async run case (optional)
Root node	*P H*	*C* (recommend)
reJoin node	*H*	*C*
Normal node		C (option for heavy processing)

Event stream

- A data packet put into the chain of processors is an event. (virtual base Event)
- Processor has the ability to processing a list of event types. (derived Event)
- Processor maintains a white list of event types which are the combination of the white list of downstream processors and itself.



Using Processor

- Processor framework is intended to be used in “Event Builder”, and make the Event building/synchronizing more flexible.

What’s more:

- “Producer”, “Data Collector” can be also be implemented as a chain of processors or just one processor.

- If link the chain of “producer”, “Data Collector” and ”Event Builder” we get a full standalone DAQ running locally.

User code / Modular binary

The line between EUDAQ core framework and user/hardware specified code

EUDAQv1

- compiled to a single shared library (so, dll).
- Device specified code (DataConverterPlugin) is inside the single library.
- Framework and user code is unfriendly mixed.
- Modification of user code will result in to re-compile all the EUDAQ lib.

EUDAQv2

A core shared library of framework and a collection of user libraries (plugins).

1. Separate user code to user folders. (done)
2. Make plugin binaries with compile-time linking. (done)
3. Make plugin binaries discovered and loaded at run-time. (TODO)
4. User derived Processor and Event complied as plugin libraries binary. (TODO)

Summary

- History: Successful story with EUDET telescope
- Target: A more common data acquisition framework
- Code status:
  - Quality code base
  - Advantage using of modern C++11
  - Open Source LGPL https://eudaq.github.io
  - Processor prototype is implemented by Richard Peschke
- Near Future:
  - Multiple "Data Collectors" (working & buggy, tested by Adrian)
  - A standard "Event Builder" for common event building case.
  - Processor framework for special Event building case. (Ongoing)
  - Modular binary (compile-time modular)
  - Interface for DAM4HEP (by Tom)

We encourage users to contribute the idea and code.