Queen Mary
University of London

DAQ For Water Cherenkov Detectors

Dr Benjamin Richards (b.richards@qmul.ac.uk)

DAQ specifics for water Cherenkov

DAQ software can sometimes be a low priority in experiment design
Many aspects of DAQ are the same no matter the experiment

However in water Cherenkov detectors:

- Our data is less dense
- More physically separated
- **Triggering** is more complex and further from hardware
- Reliability is very important
- **Large data buffers** are required

My Prerequisites

Find a DAQ framework that could provide the specific qualities for Cherenkov detectors as well as being **modular**, **easily scalable** with large customizability and **fault tolerance**

- Hardware test stands (few PMTs)
- ANNIE experiment (30-200 PMTs)
- E61 Experiment (~7,500 PMTs [365 MPMTs])
- Hyper-K experiment (~40,000 PMTs)

Unfortunately I couldn’t find a good fit.

ToolDAQ

ToolDAQ is an open source DAQ Framework developed in the UK.

It was deigned to incorporate the best features of other DAQ whilst:

1. Being very easy and fast to develop DAQ implementations in a very modular way.
2. Including dynamic service discovery and scalable network infrastructure to allow its use on large scale experiments.

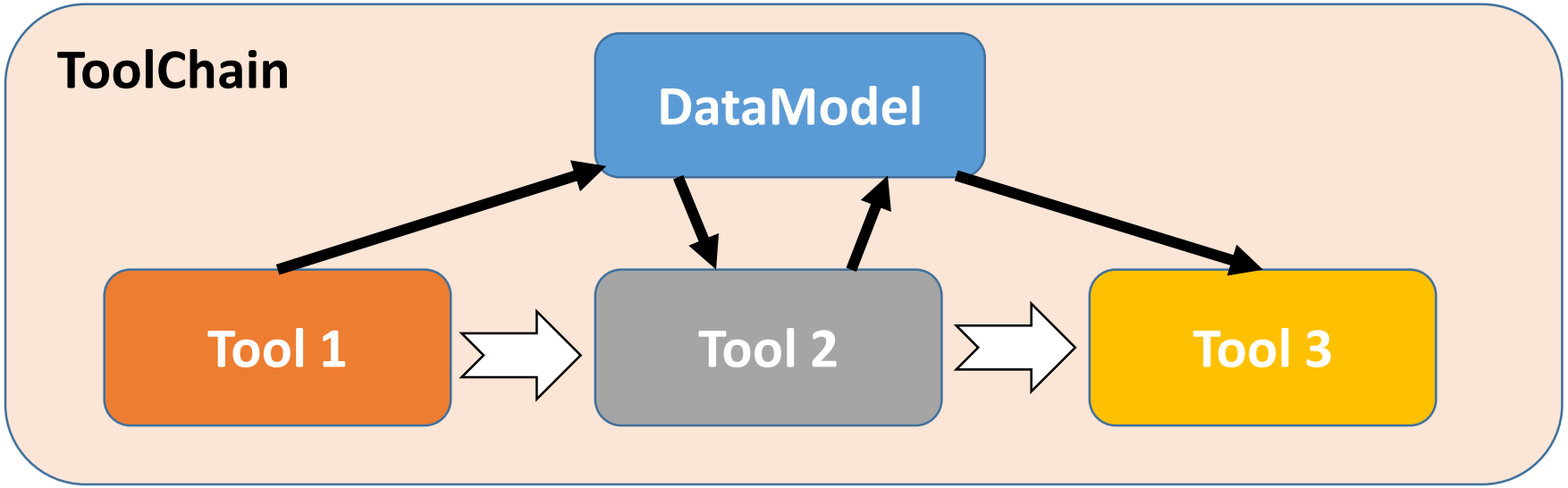
Features

- Pure C++
- Fast Development
- Very Lightweight
- Modular
- Highly Customisable / Hot swappable modules
- Scalable (built in service discovery and control)
- Fault tolerant (dynamic connectivity, discovery, message caching)
- Underlying transport mechanisms ZMQ (Multilanguage Bindings)
- JSON formatted message passing
- Few external dependencies (Boost, ZMQ)

Points of focus

- **Modularity / Customisability**
- **Scalability / Dynamisms**
- **Fault Tolerance / Error Correcting**

ToolDAQ Modularity and Customisability In Structure

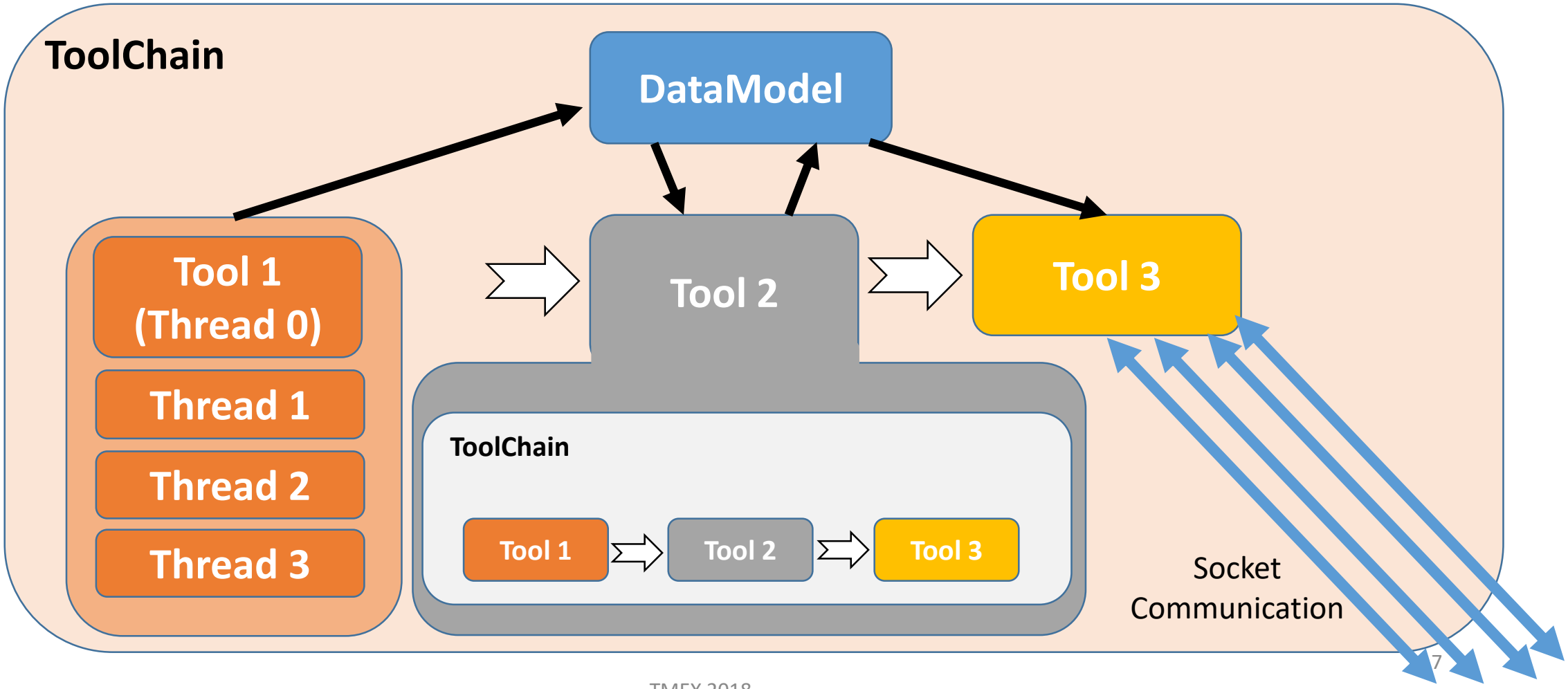


Tool = Modular classes that make up your program

ToolChain = Class that holds the modular Tools

DataModel = Shared / transient data class. Any object/variable/instance in the DataModel class is shared between all tools

Tools

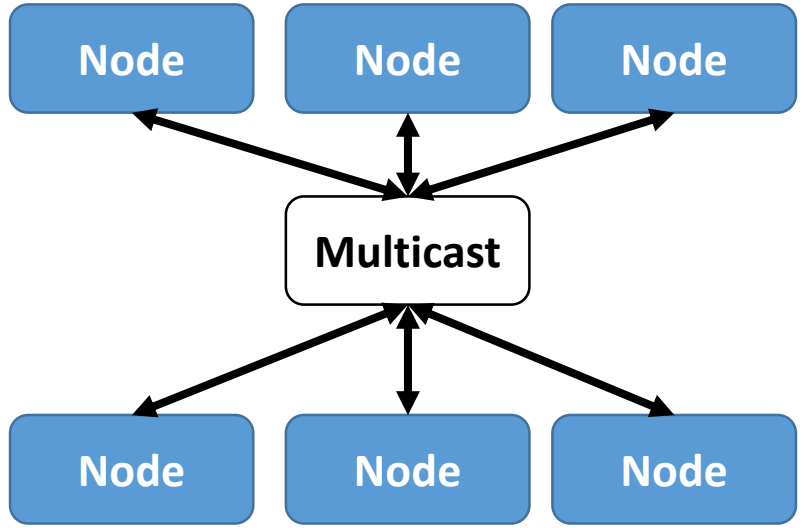

ToolChain
DataModel
Tool 1
(Thread 0)
Thread 1
Thread 2
Thread 3
Tool 2
Tool 3
ToolChain
Tool 1
Tool 2
Tool 3
Socket
Communication

Dynamic Service Discovery

The core Framework has multiple threads that run both in the ToolChains and NodeDaemons that take care of all the control systems, service discovery, etc…

Dynamic service discovery lets every single Node Daemon, ToolChain and service know about each other via use of multicast beacons.

This is how remote control is achieved anywhere on the network



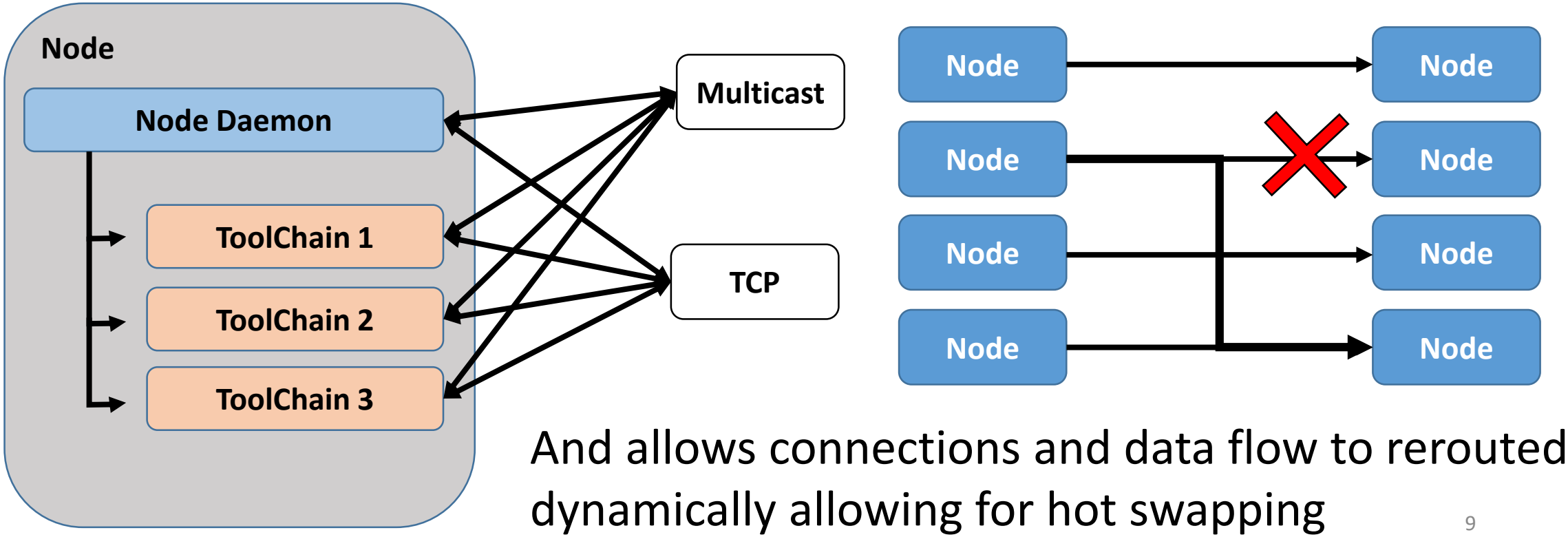
[UUID, Name , IP, Service, Port, Status, Timestamp]

Distributed Node Management & Hot Swapping

Most DAQ systems will require multiple distributed nodes

Each can have multiple ToolChains running on them

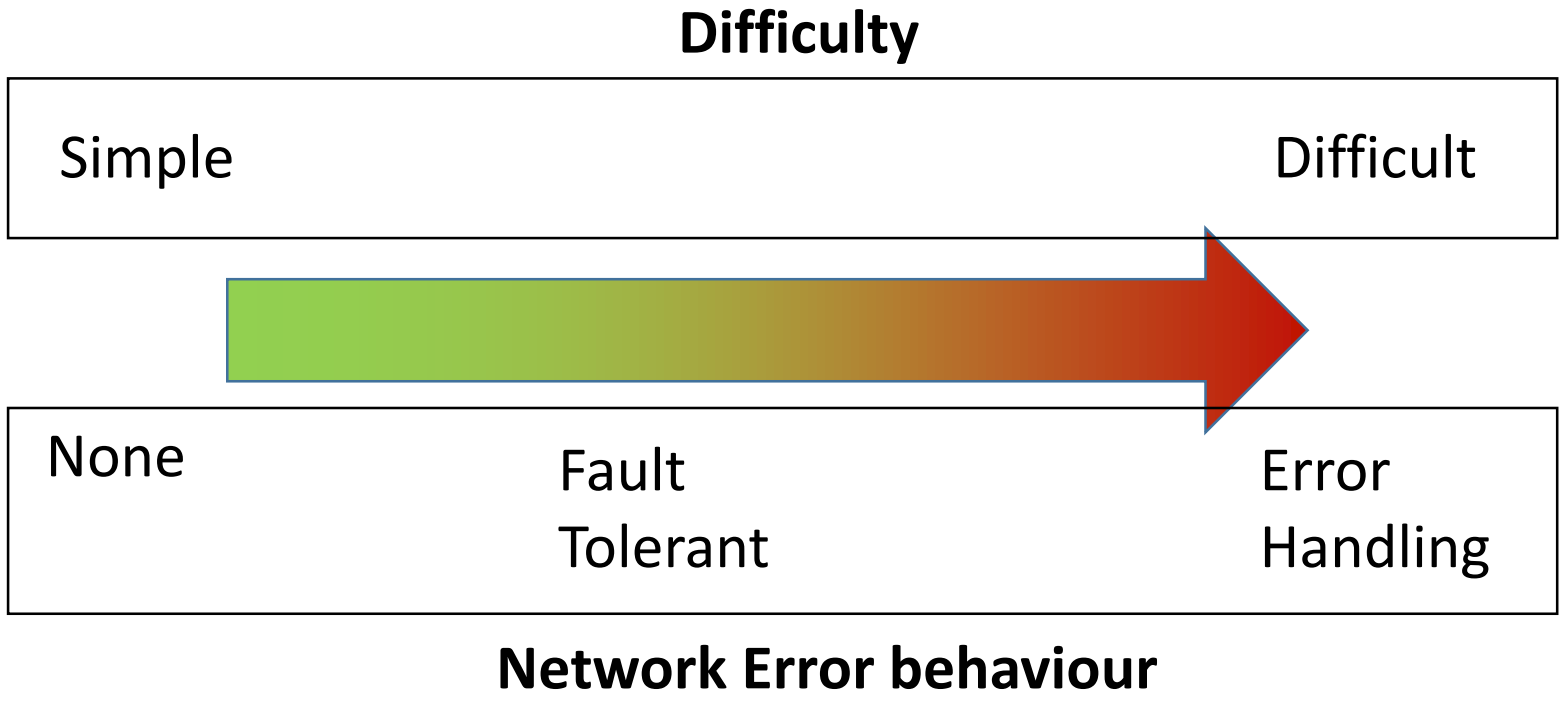
So ToolDAQ has a node control and monitoring system



And allows connections and data flow to rerouted dynamically allowing for hot swapping

Fault Tolerance And Error Correcting

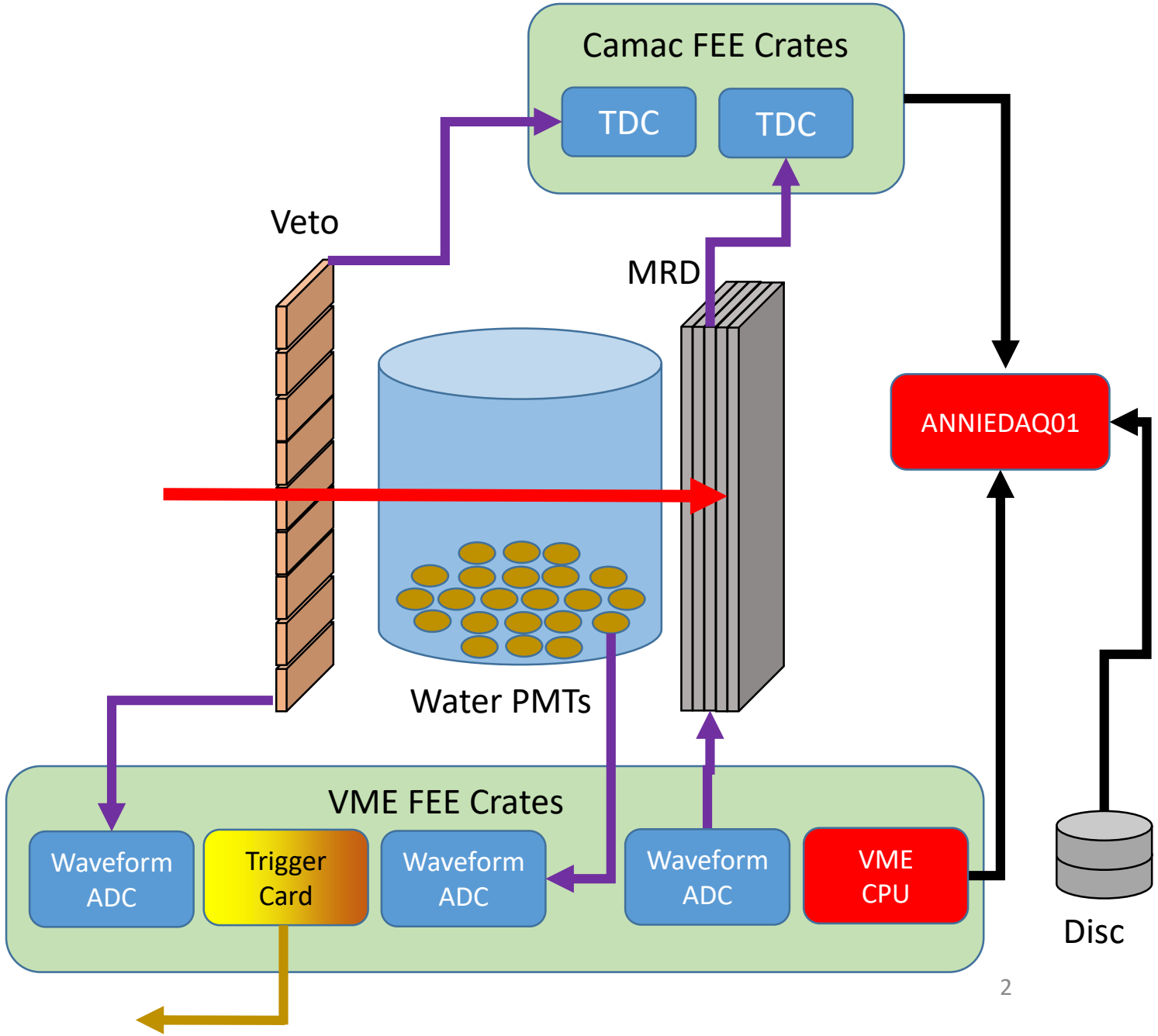
- ToolDAQ makes use of ZMQ to provide a fault tolerant scalable messaging
- Use of ZMQ, message buffering and Service discovery allows for creation of DAQ that can not just be fault tolerant but handle errors.



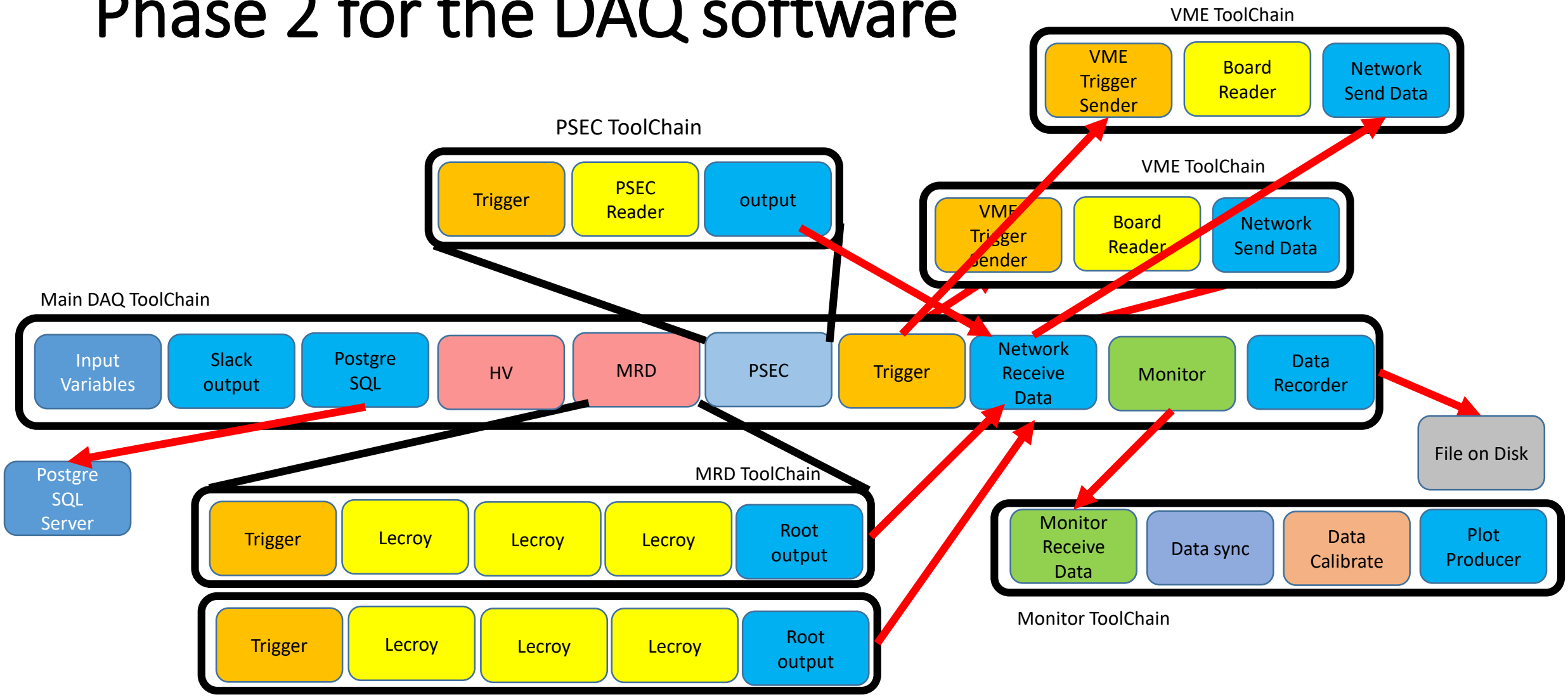
ANNIE

- Multiple asynchronous data sources MRD, Veto, PMTs LAPPDs
  - ADC Koto Boards
  - Camac TDC
  - PSec LAPPDs
  - Trigger stream
- Fault tolerant
- Flexible to changes in data and trigger

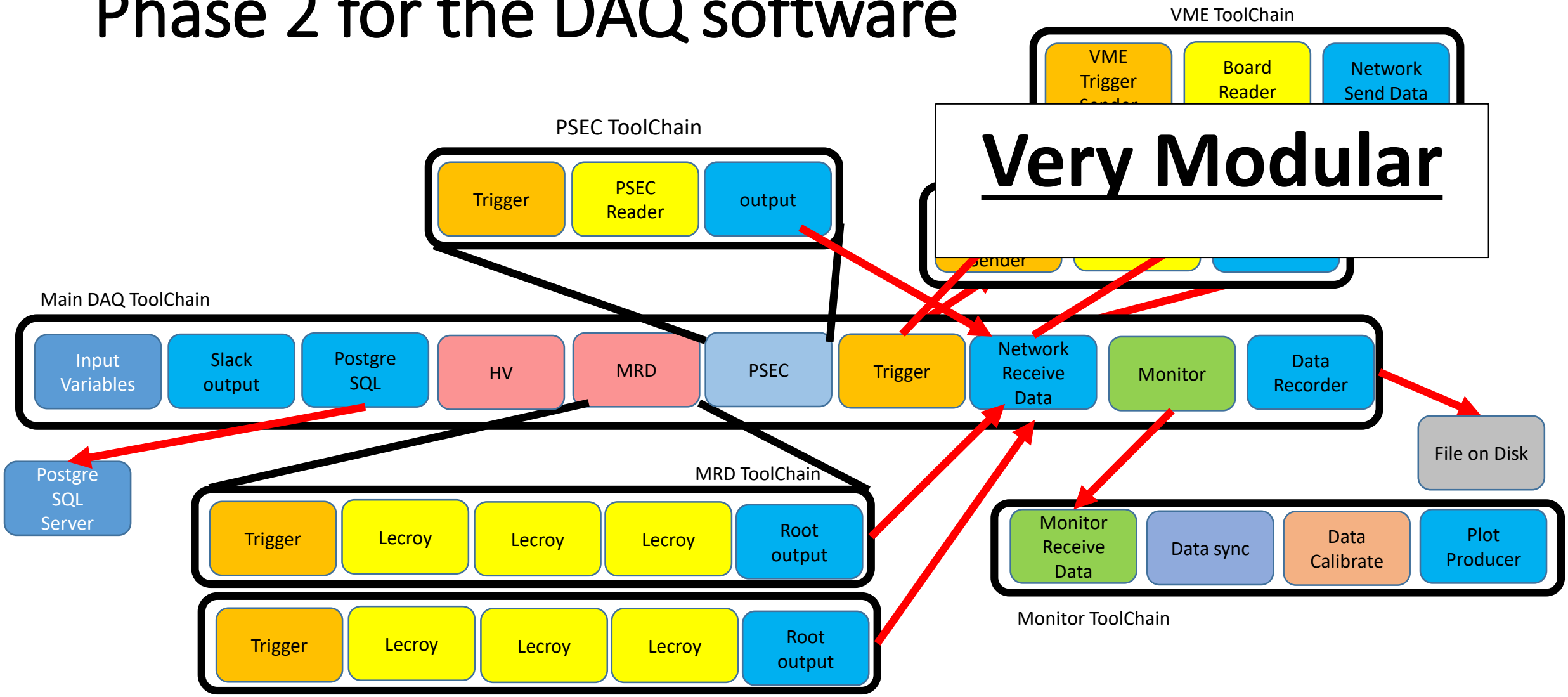
- Also used for analysis/reconstruction



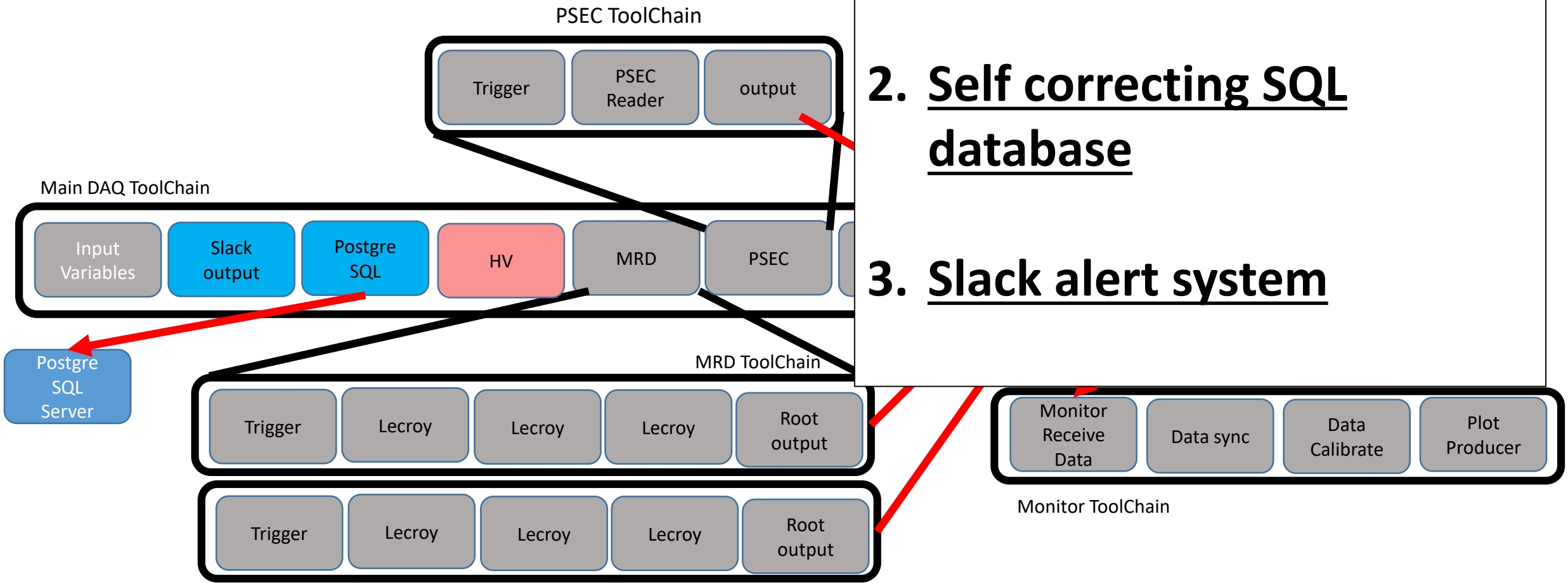
Phase 2 for the DAQ software



Phase 2 for the DAQ software



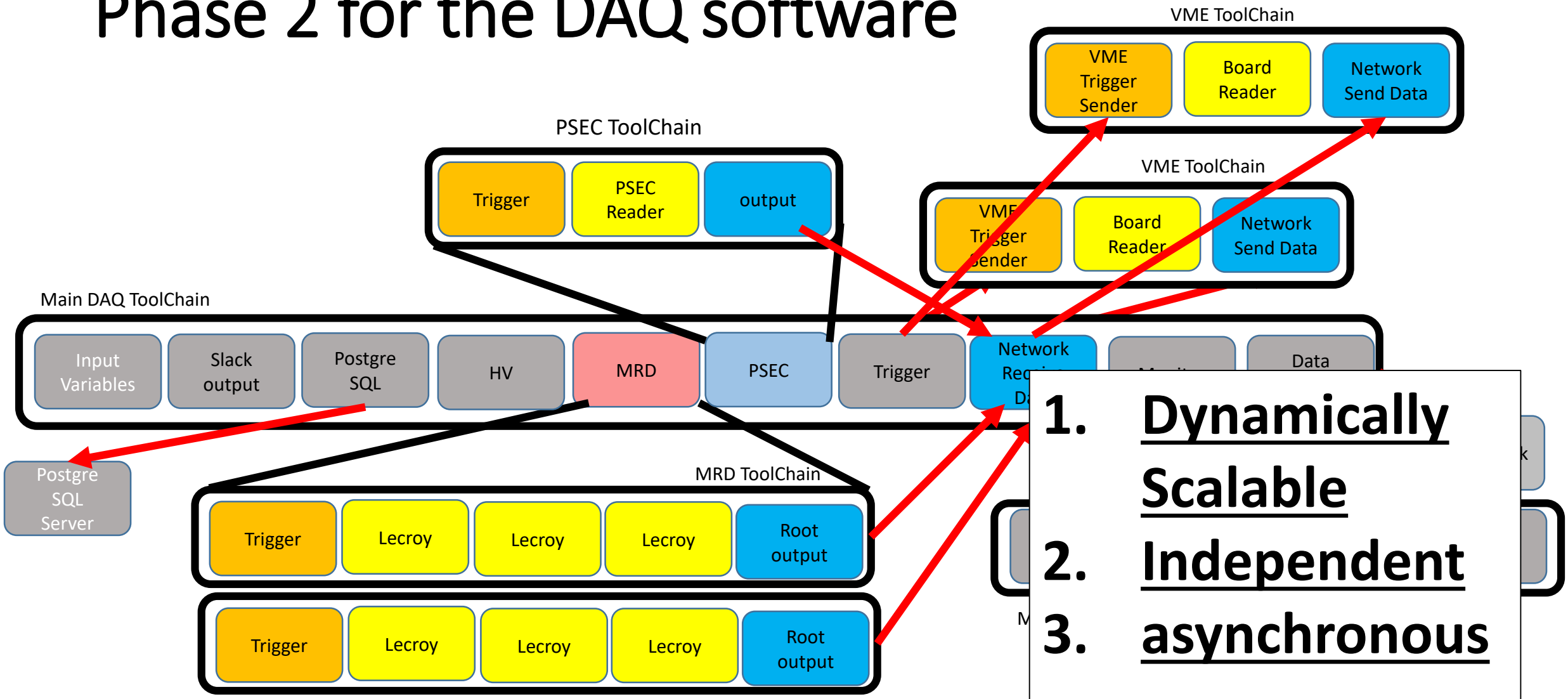
Phase 2 for the DAQ softw



Nice features

1. **Integrated HV slow control**
2. **Self correcting SQL database**
3. **Slack alert system**

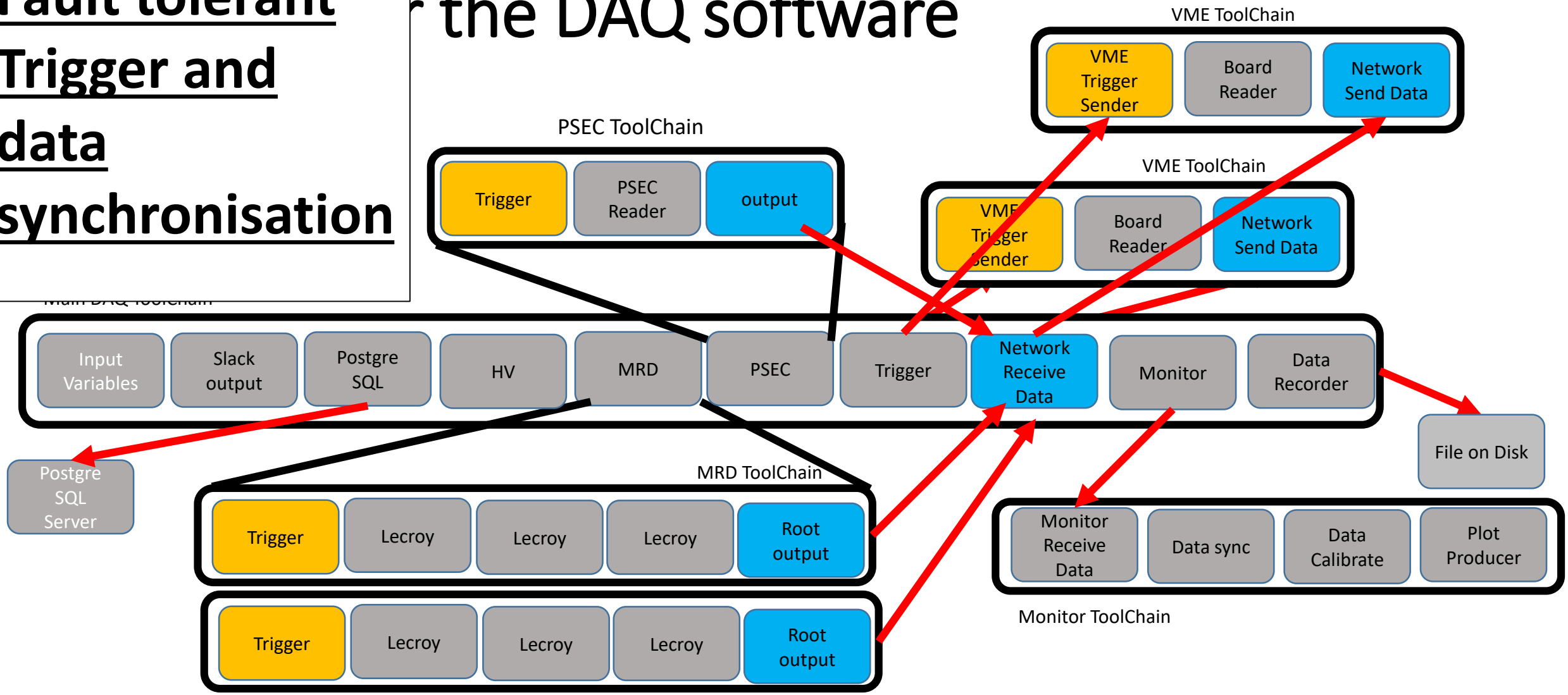
Phase 2 for the DAQ software



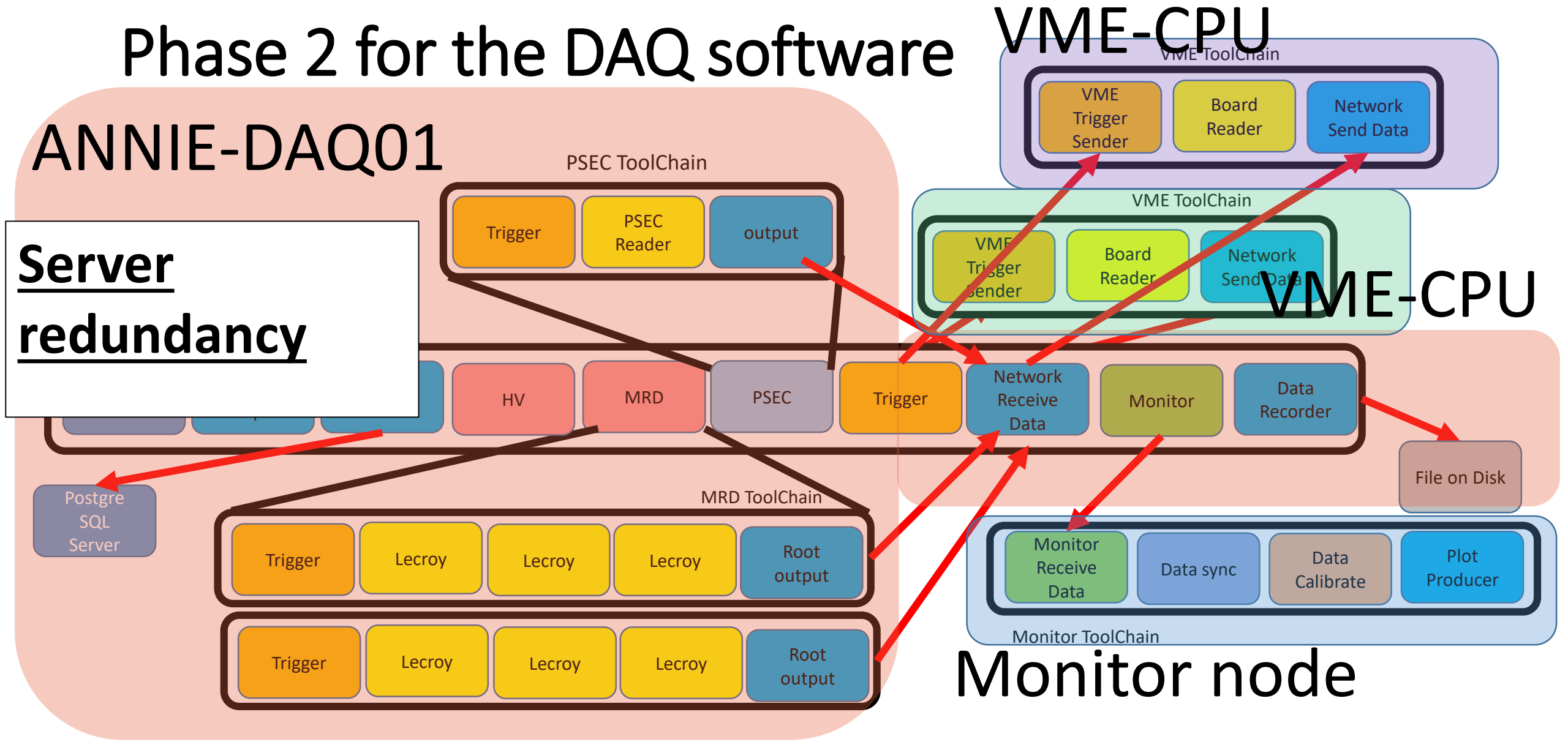
1. **Dynamically Scalable**
2. **Independent**
3. **asynchronous**

Fault tolerant Trigger and data synchronisation

r the DAQ software



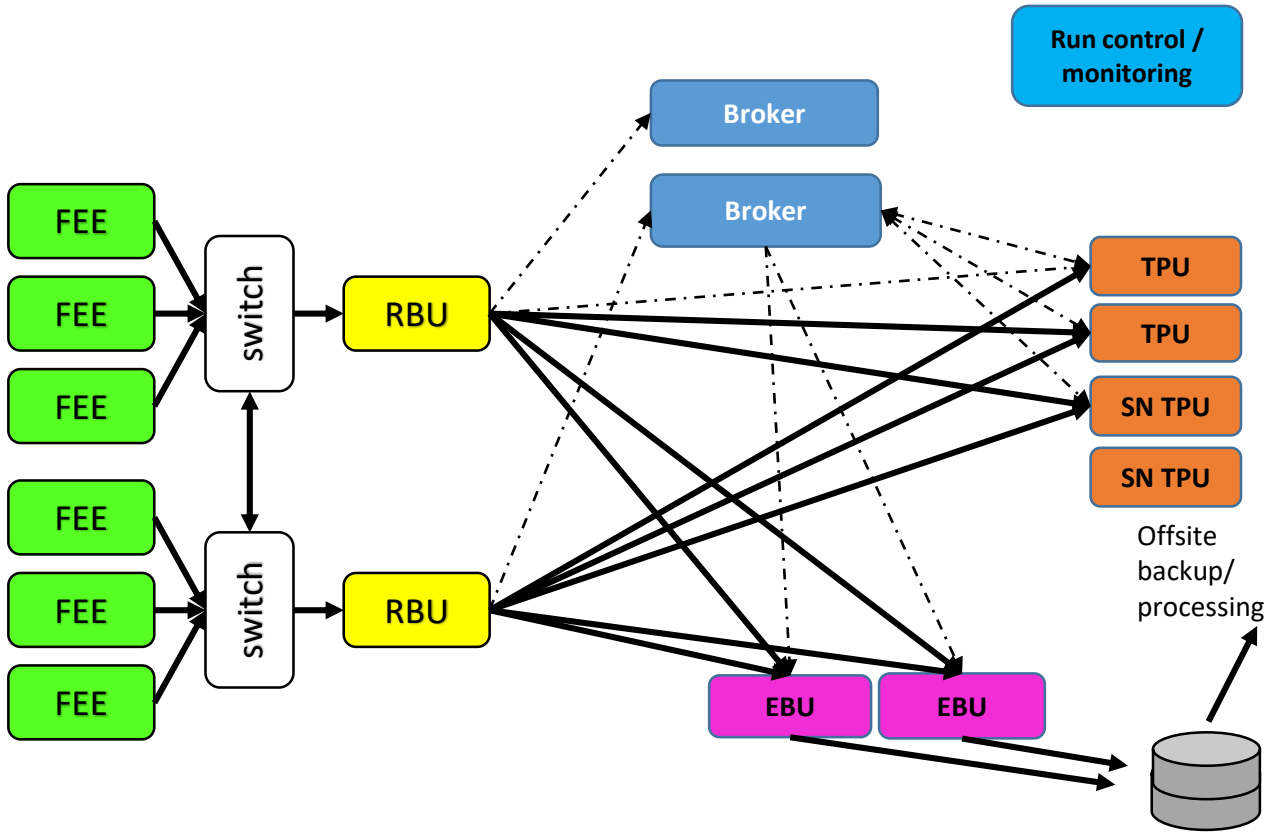
Phase 2 for the DAQ software

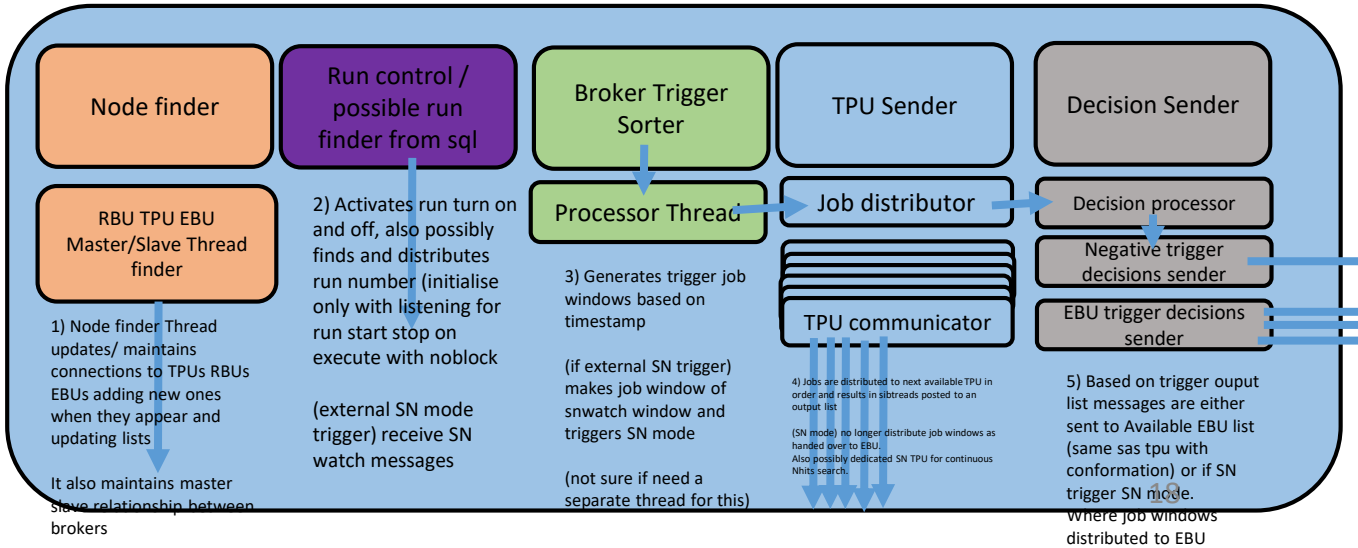


Hyper-K

- Larger Beast
  - 40,000 channels
  - 2000 FEEs
  - 150 computing nodes
  - Dead timeless
  - Separate GPU Trigger farm
  - Readout buffering
  - Event Building

- Self maintaining
- Highly fault tolerant

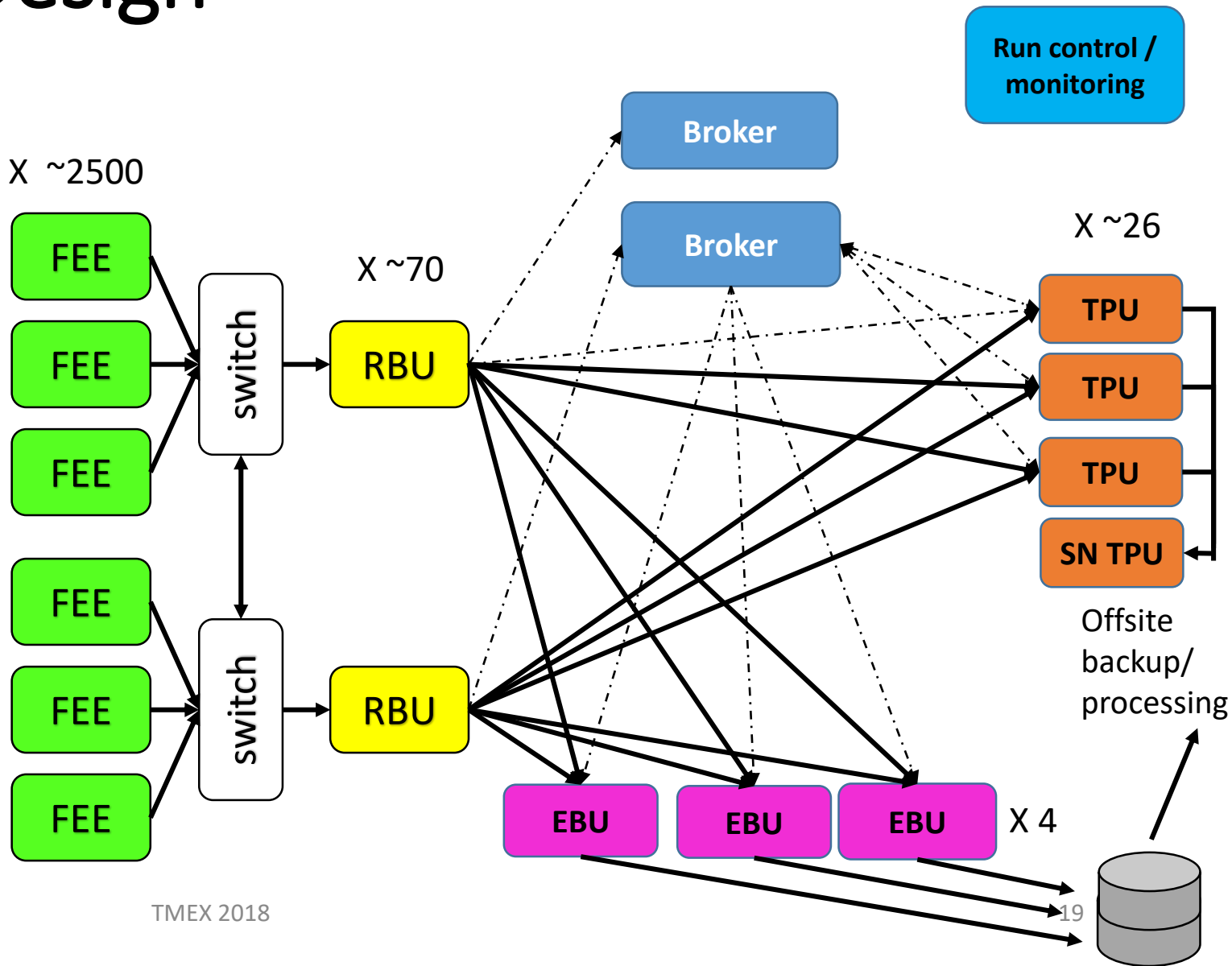




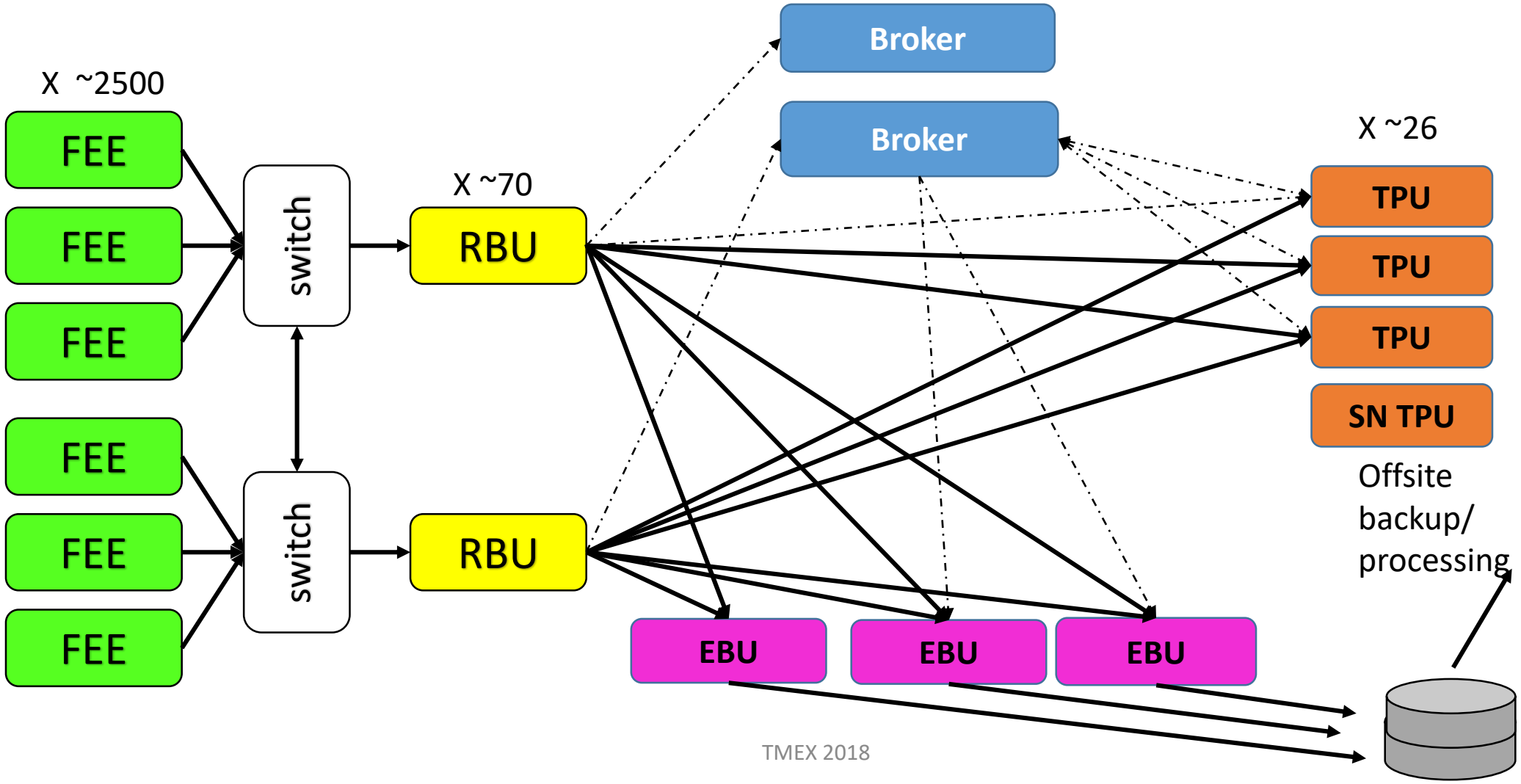
Current HK DAQ Design

Features:

- FEEs can be rerouted to be read out by any RBU
- Broker assisted communication between RBU TPU and EBU
- Master slave Broker redundancy
- Dynamic routing and fault tolerance (hot swapable)
- Ability to switch on and off parts of the detector from the data stream
- TPU farm (GPU based)
- Supernova buffer/Readout
- Centralised run control and monitoring/logging



Hyper-K Design



Summary

- Watcher Cherenkov detectors have specific challenges to overcome
- Distributed buffering to solve large data buffering issues
- Intelligent distributed triggering (see next talk for algorithms)
- Reliability can be mitigated with redundancy and fault tolerant communications
- Scalability can be addressed with dynamic service discovery and scalable network layers