

# Data Processing for CLAS12

---

Gagik Gavalian (Jefferson National Laboratory)  
EIC Streaming Readout, May 22 2019, Camogli

# Introduction

## CONVENTIONAL WISDOM

- If things worked well 20 years ago, we should follow the same path.
- Analysis tools that we used for past 30 years are just fine, why reinvent ?
- The data we are writing we can analyze with our existing tools, so what if it's 1000 times larger data ?

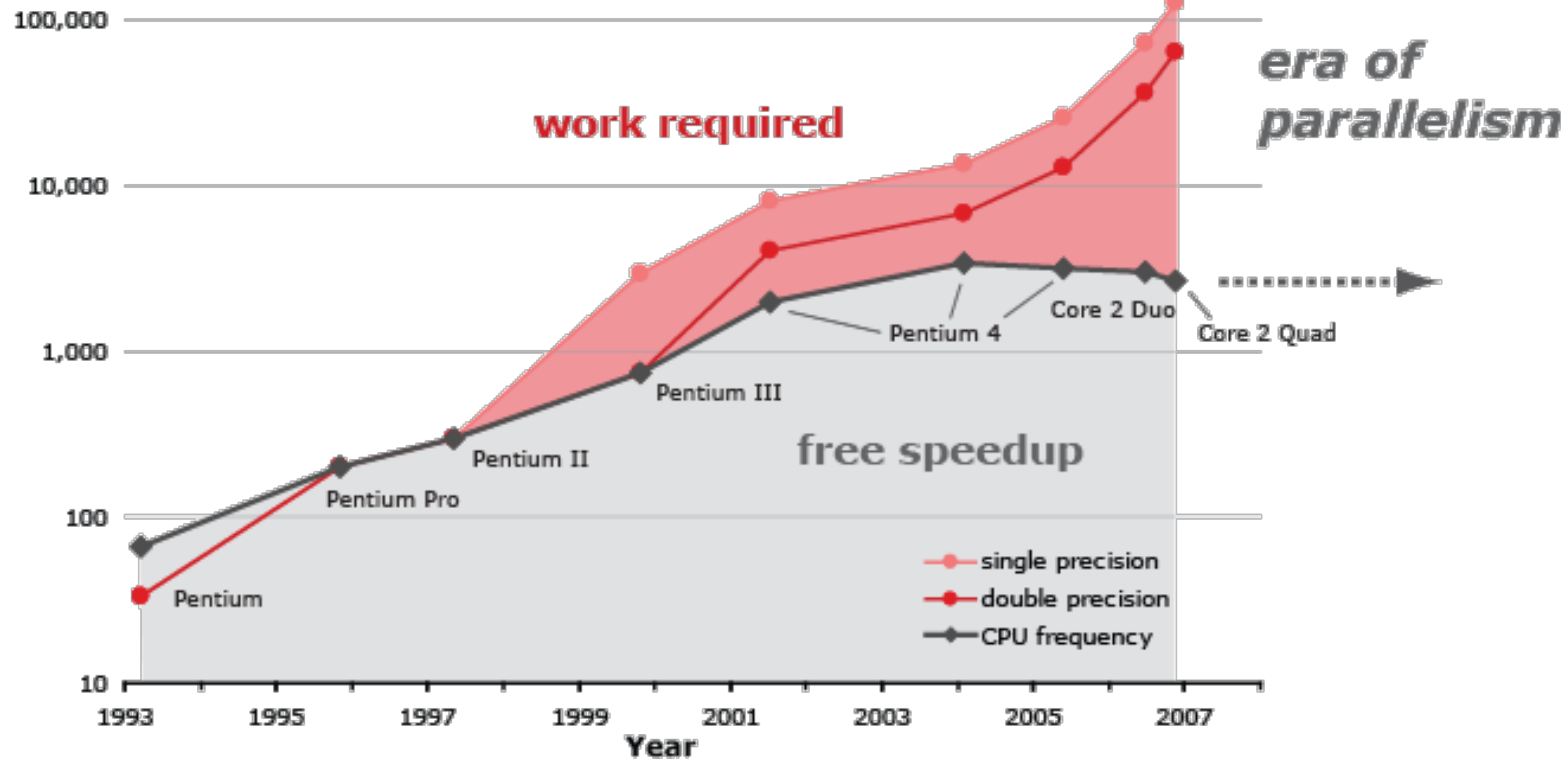
## QUESTIONS

- Is computing evolving the same way it was 20 years ago ?
- Software architecture models have to change ?
- Do we need new approaches to data storage and distribution ?

# Evolution of Computing

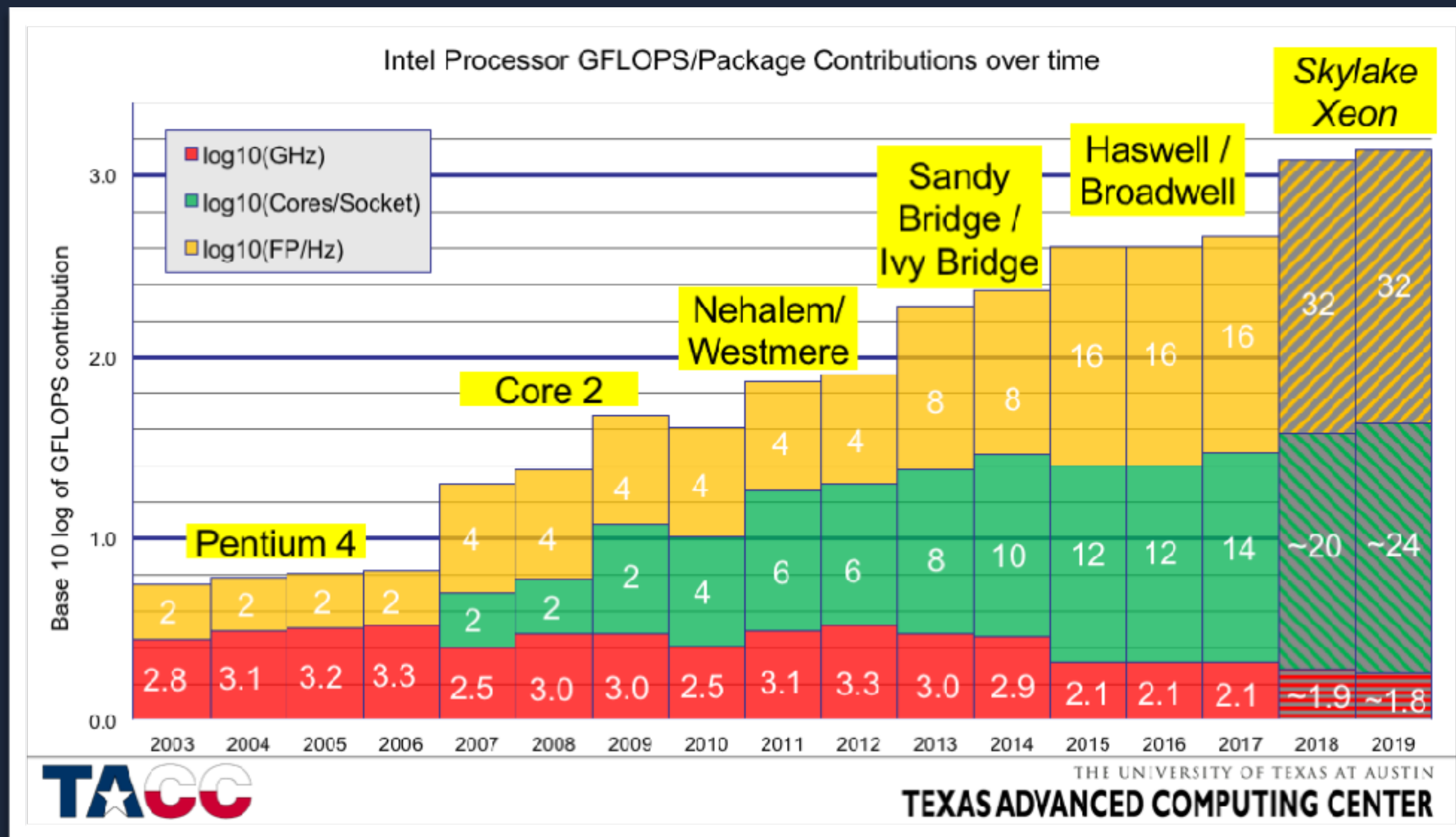
## Evolution of Intel Platforms

Floating point peak performance [Mflop/s]  
CPU frequency [MHz]



data: [www.sandpile.org](http://www.sandpile.org)

# Evolution of Computing

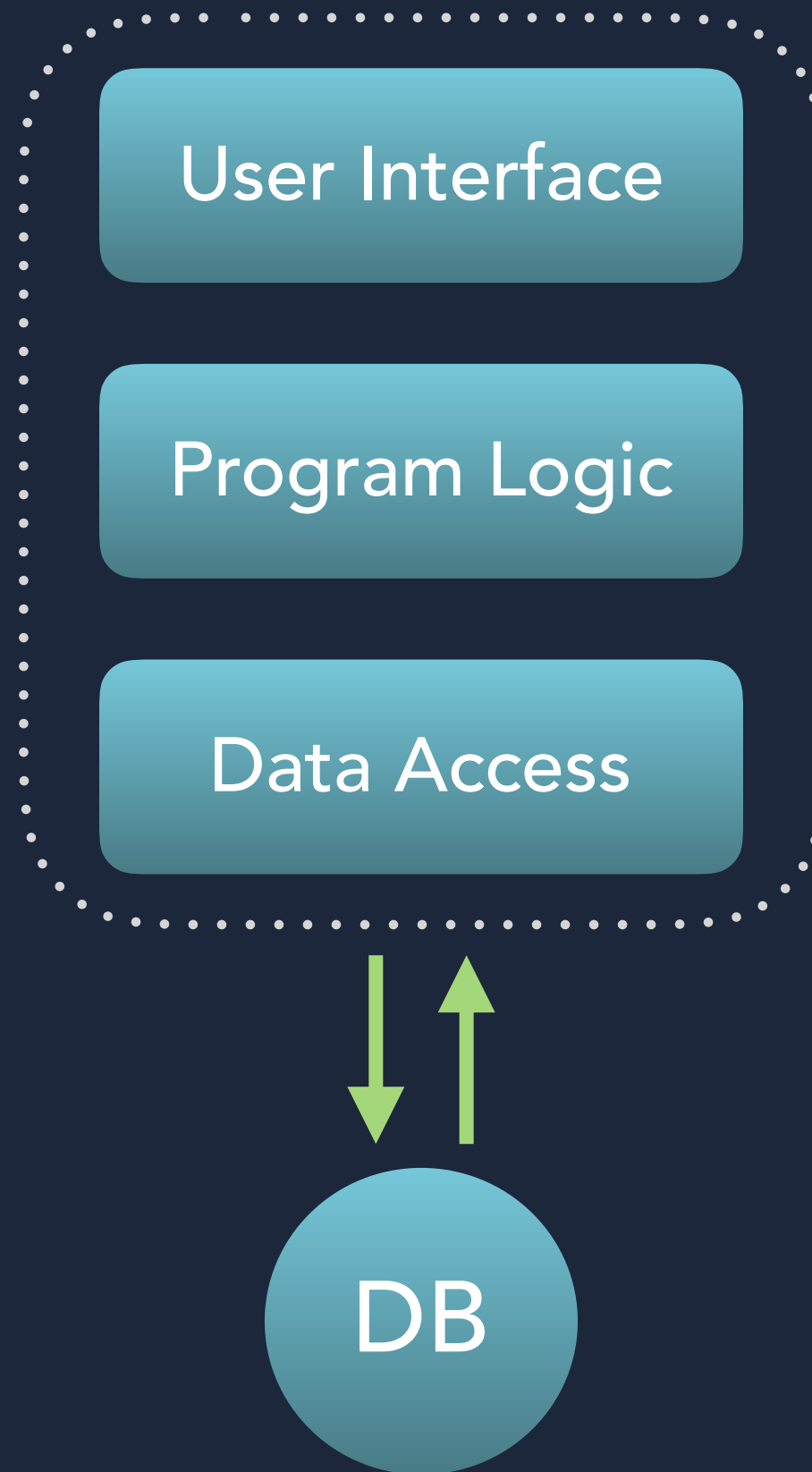


- Major software engineering methods and tools currently focus on sequential software development.
- However, every developer is now confronted with parallel programming applications such as server applications.
- If application is not parallel performance can not be improved by additional cores.
- As more cores are integrated on the same chip, clock rates may decrease, sequential applications will be slower with new processor generation.

- The whole spectrum of software engineering – from design over testing to maintenance – has to be revisited in the light of parallelism. Nondeterminism adds a new dimension of complexity.
- Many parts of the existing software stack require revisions, each providing challenges and opportunities for parallelism exploration.

# Service Oriented Architecture (SOA)

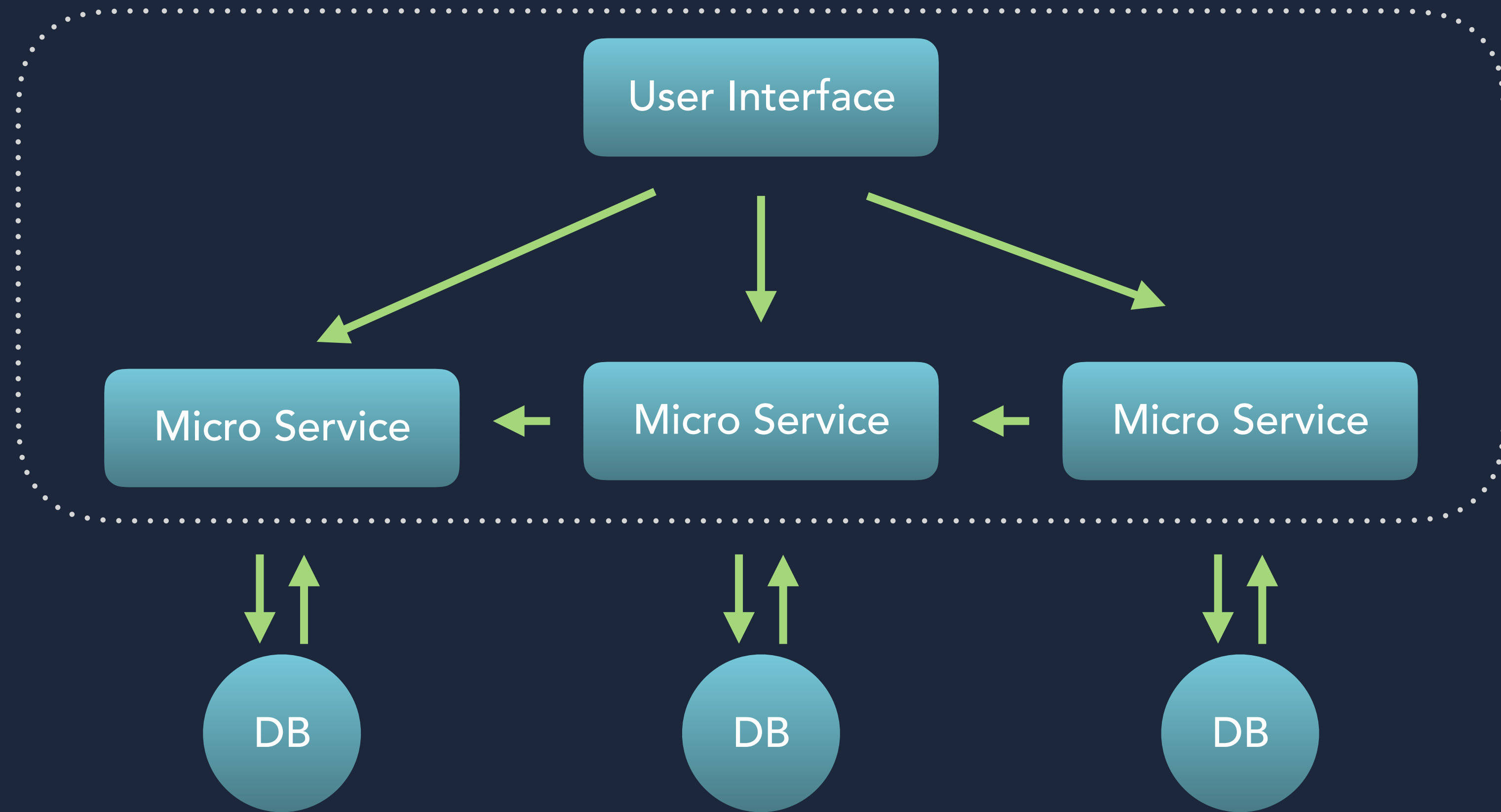
## MONOLITIC ARCHITECTURE



### Monolithic Approach

Provides a program based on the libraries (API) to accomplish a specific task. If different task had to be performed one has to re-write the program.

## SOA ARCHITECTURE

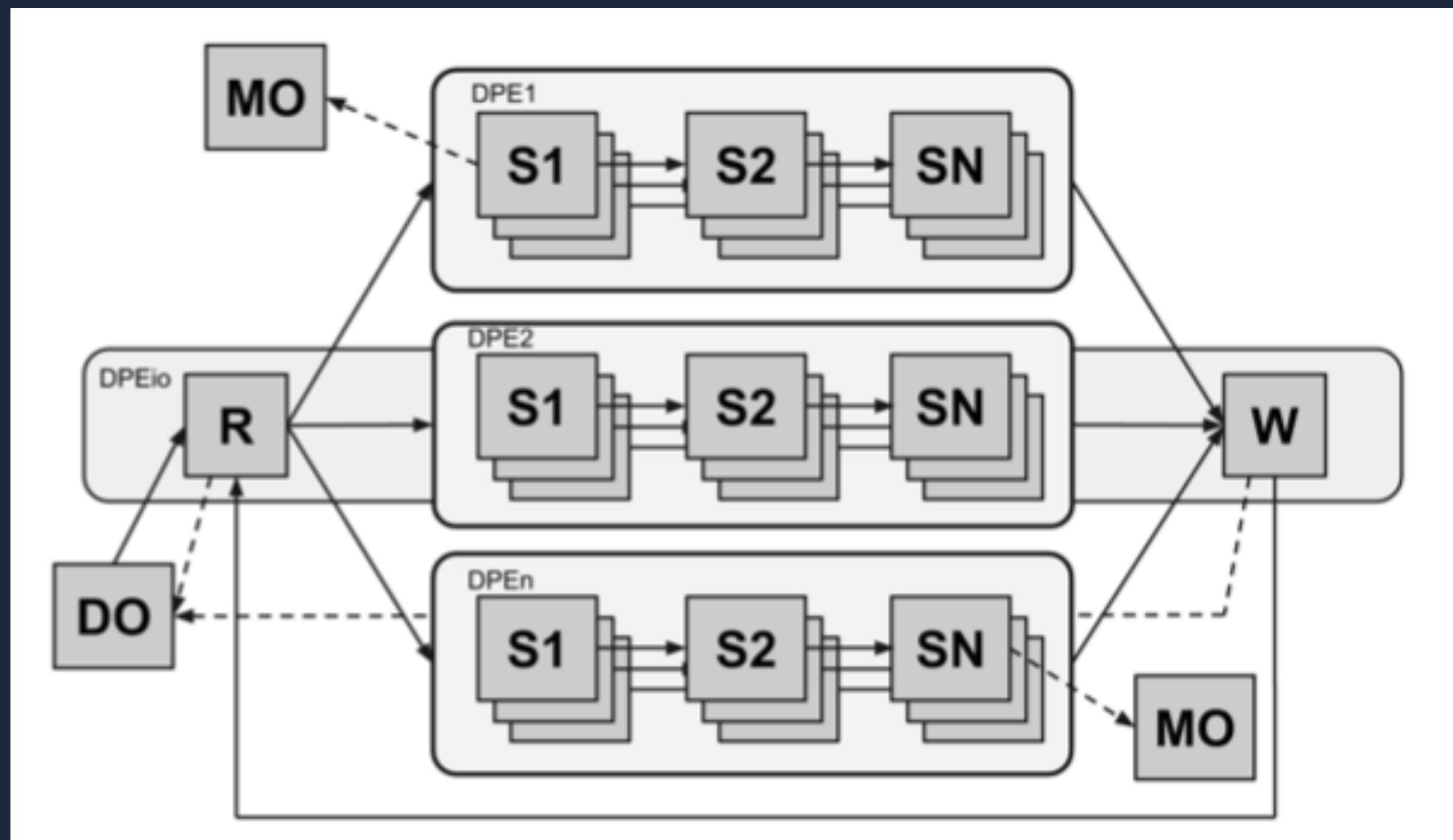


### SOA Approach

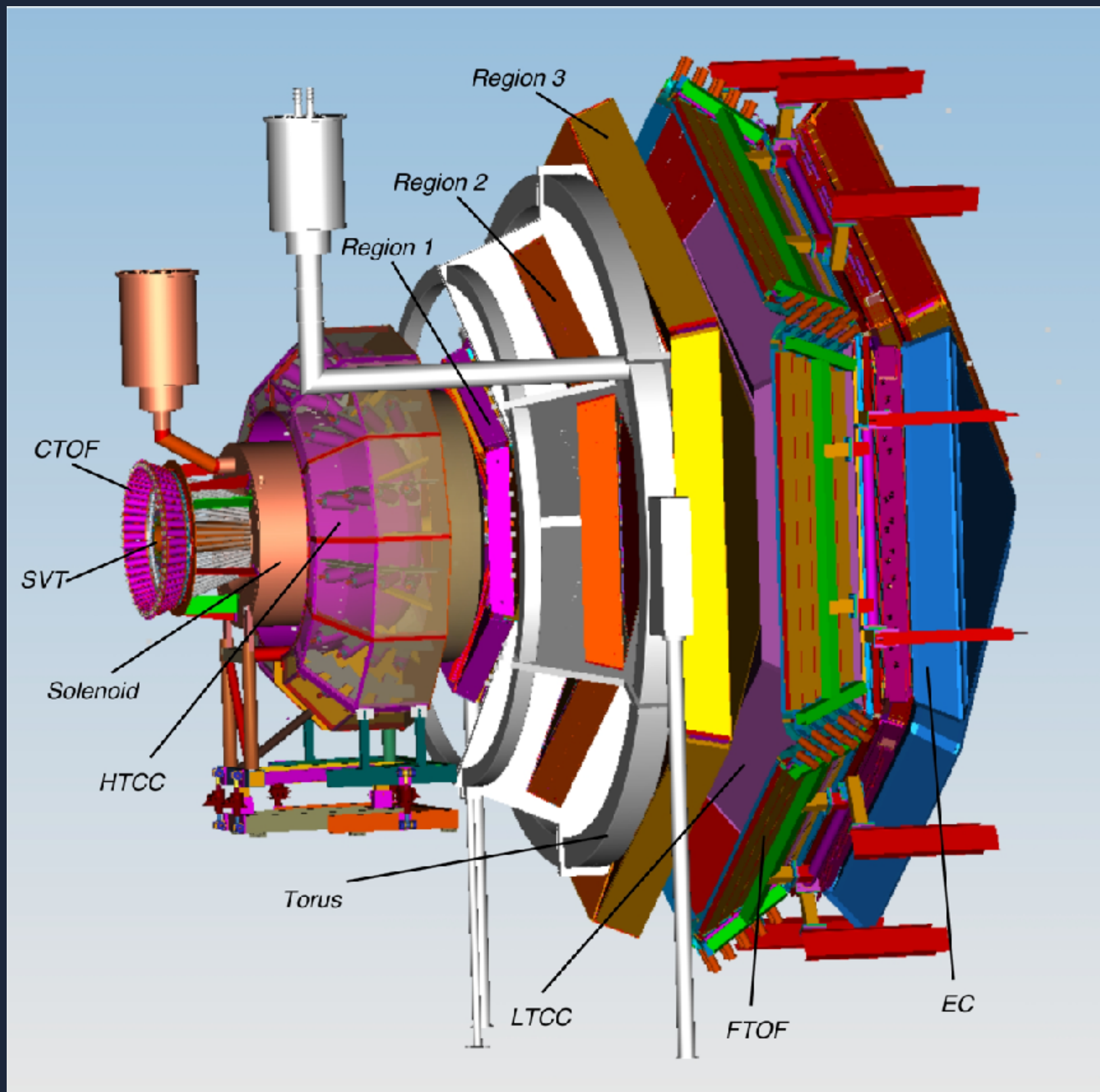
Provides Services that can do one specific task. User can build an application by chaining services, and passing data through the chain and accomplishing different tasks.

# Service Oriented Architecture (SOA) (<https://claraweb.jlab.org/clara/>)

- CLARA Framework aims to enhance the efficiency agility and productivity PDP processes.
- CLARA is an approach to develop data processing applications based on the concept of multiple asynchronous processes (Services)
- The Application in CLARA is system of data streams being transformed by multithreaded services.
- Service Composition is comprised of services that have been assembled to provide functionality required to accomplish a specific data processing task:
  - Data Reconstruction
  - Event filtering
  - Kinematic Fitter
- CLARA makes a clear separation between the service programmer and the data application designer.



# CLAS 12 Detector



## DETECTOR COMPOSITION:

- Drift Chamber inside Toroidal field for forward tracks.
- Electromagnetic Calorimeter for electron identification and neutral particle detector.
- Time of Flight system for particle identification.
- High Threshold Cherenkov Detector for electron pion rejection.
- Silicon tracker for central detector charged particle tracking in Solenoidal Field.
- Central Neutron Detector for neutron identification.

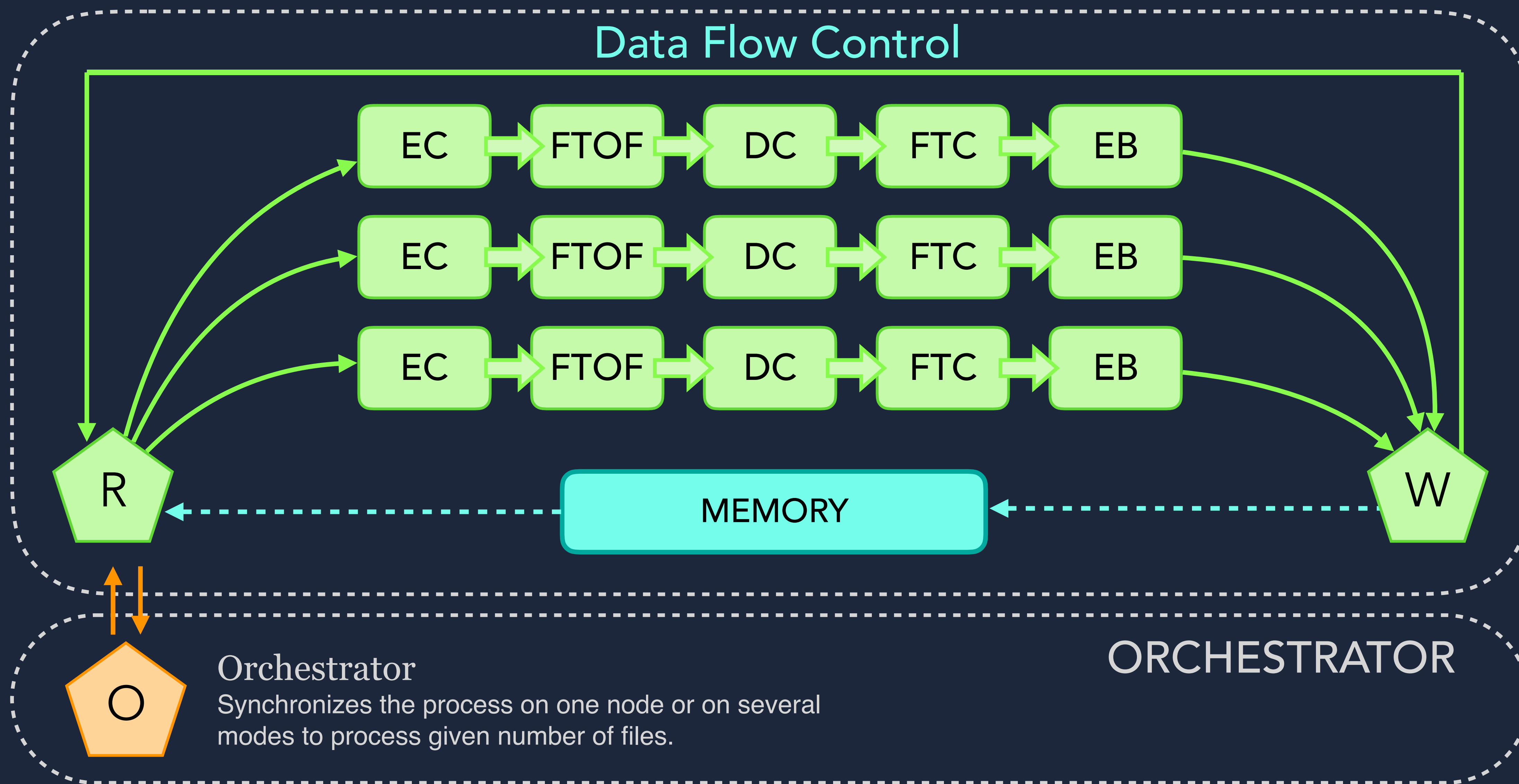
## DATA ACQUISITION:

- >100K Channels
- DAQ data rate 12 kHz,
- Data rate 400 Mb/sec
- Up-to-Date collected ~1.2 Pb

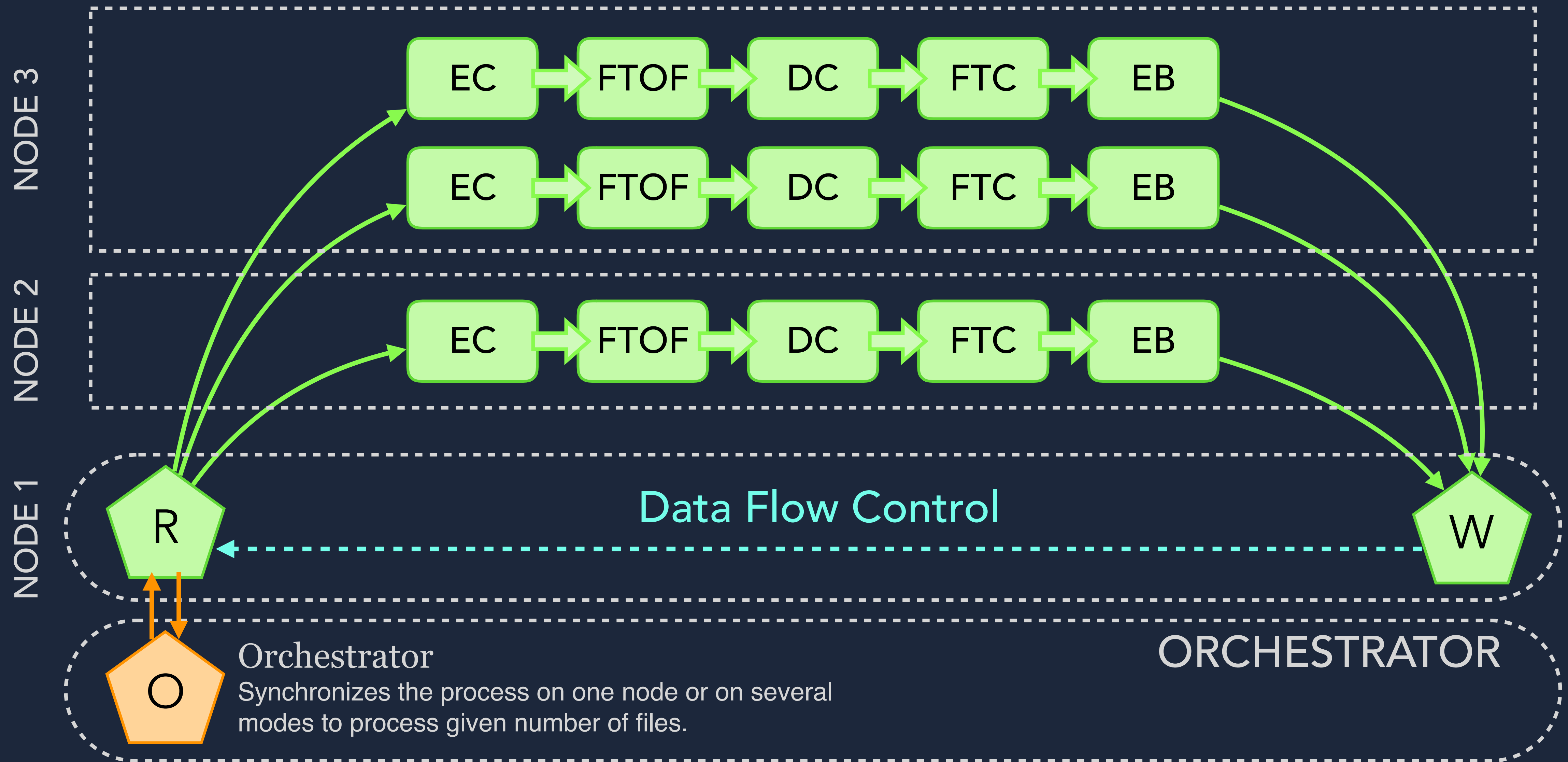
# CLARA Data Processing

## DATA PROCESSING ENVIRONMENT (DPE)

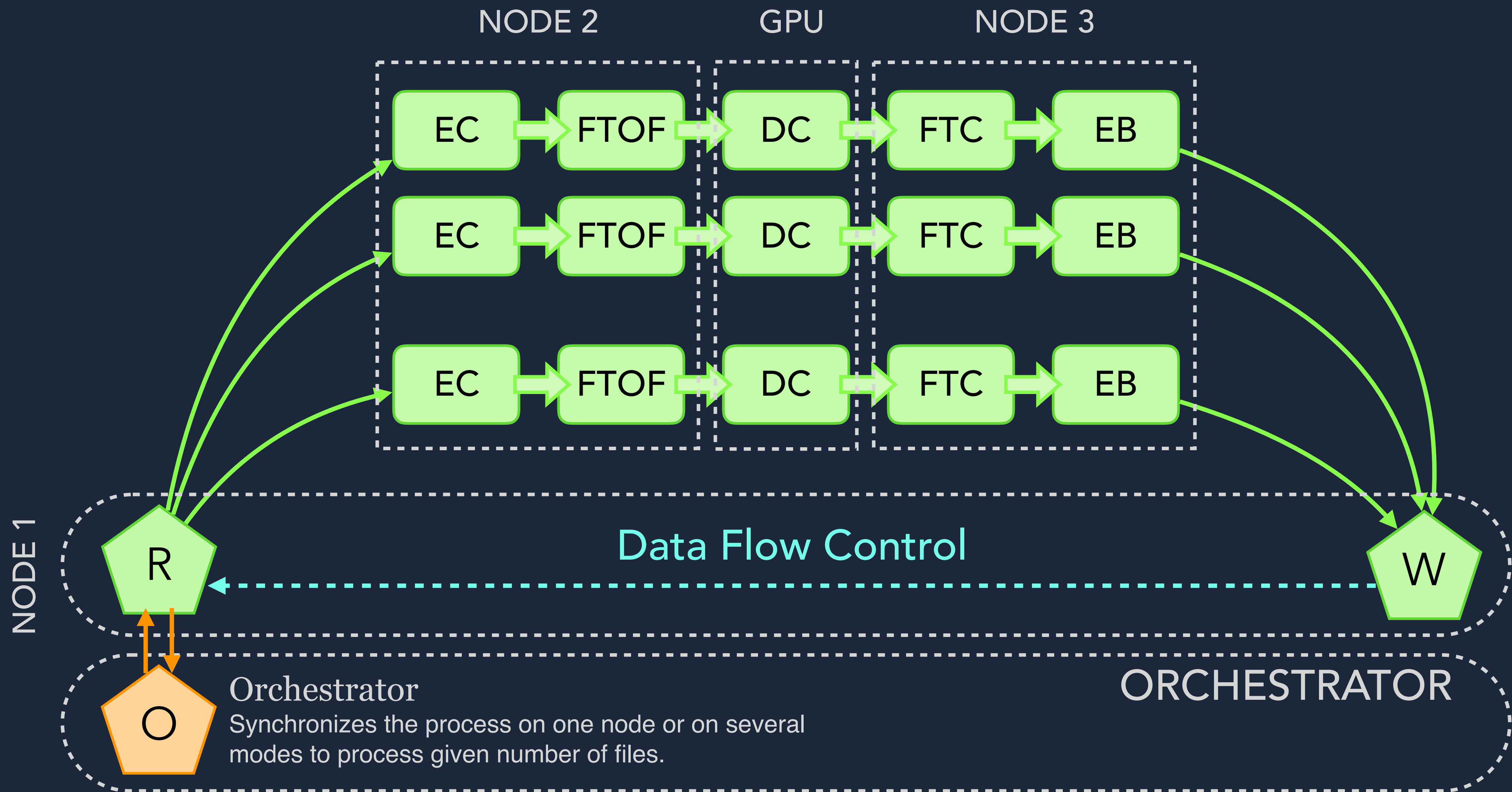
### Data Flow Control



# CLAS12 Data Processing (Vertical Scaling)



# CLAS12 Data Processing (Multi-Platform Scaling)



# CLAS12 Data Processing

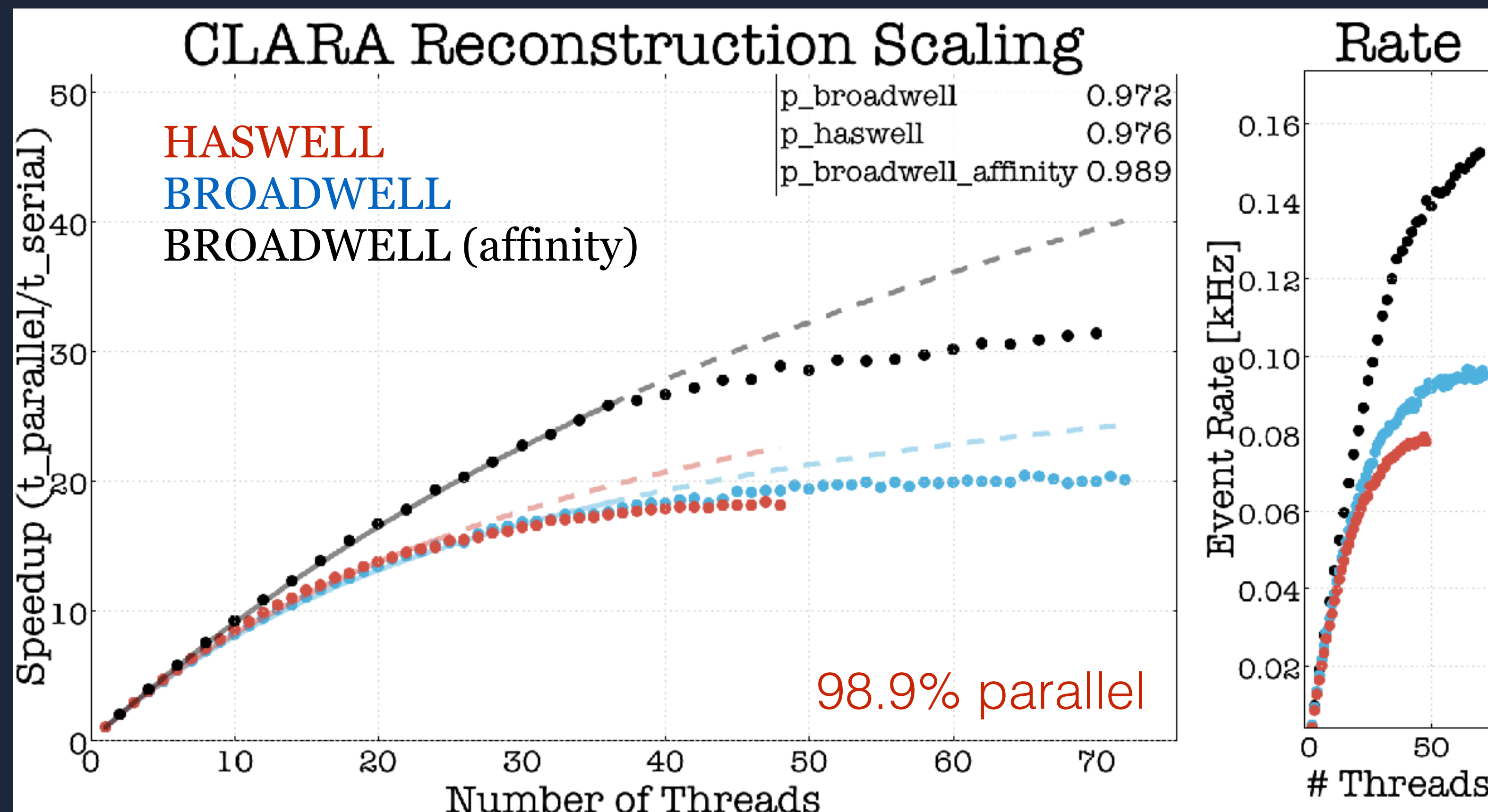
## Amdahl's law

gives the theoretical speedup in latency of the execution of a task at fixed workload that can be expected of a system whose resources are improved.

$$S_{\text{latency}}(s) = \frac{1}{(1 - p) + \frac{p}{s}}$$

$p$  - is the percentage of the execution time of the whole task concerning the part that benefits from the improvement of the resources of the system before the improvement.

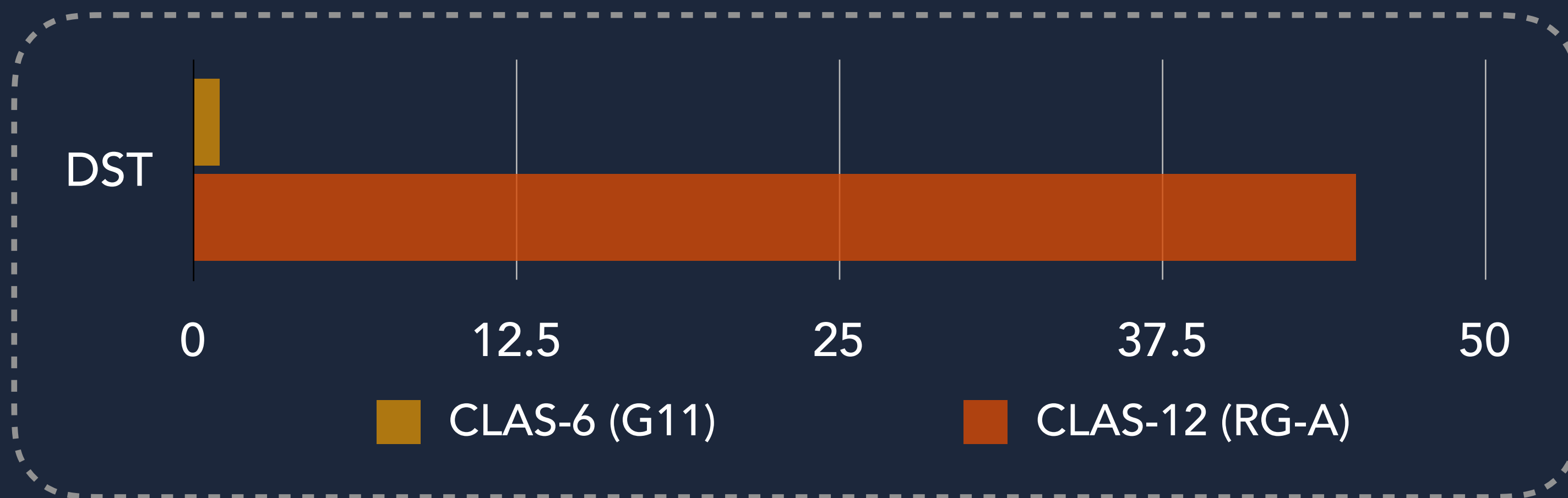
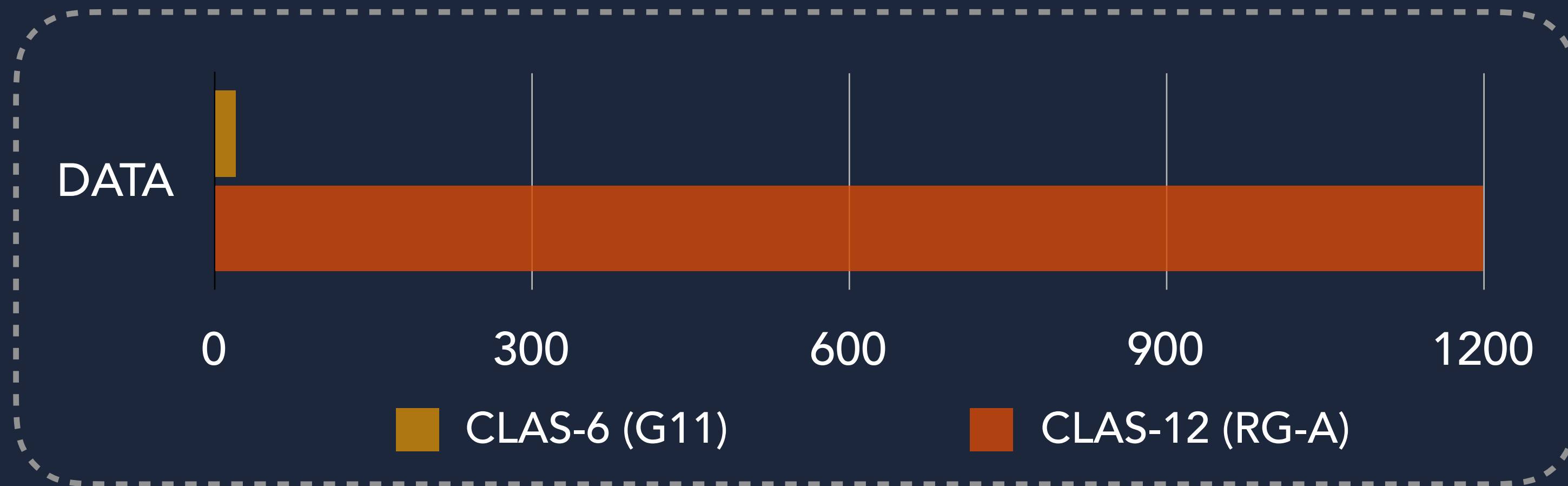
$s$  - is the speedup in latency of the execution of the part of the task that benefits from the improvement of the resources of the system



## Thread Affinity

Operating system forces the thread to run on one specific core without changing to the available idle cores. Switching requires data to move along with the process causing slow down.

# CLAS 12 Detector vs CLAS



## DATA SIZES:

- CLAS12 First Experiment collected **~50x** more data than longest experiment in CLAS6 era (**~1.12 PB**)
- Data Summary tapes are **~100x** larger (**45 TB**) than G11 experiment (on the graph compressed size is shown)

# CLAS 12 Data Format

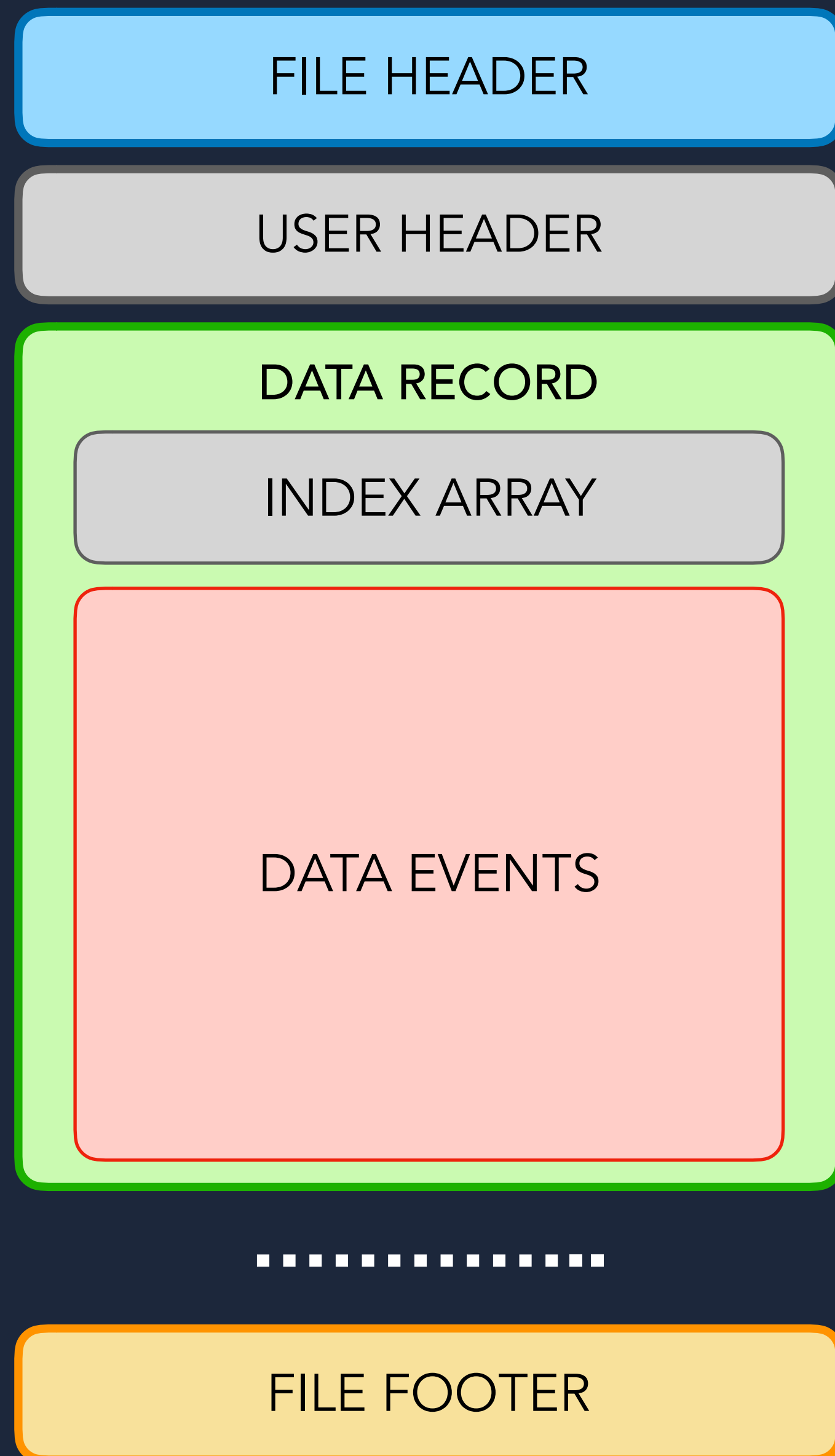
## Software Architecture:

- CLAS12 Software is in JAVA. We need data format that has read/write from JAVA (ROOT is not an option).
- Industry has many data formats for JAVA (Apache Avro, LCIO, HDF5).

## Existing Formats:

- EVIO was used in DAQ (has C++/Java interface) and was decided to be used in reconstruction
  - No compression
  - No random access
  - No support for large (>2GB) files (in Java API)
  - No efficient chunk reading (can bring down LUSTRE disks)
- ROOT is widely used in final physics analysis:
  - No Java interface

# HIPO Data Format (File Format)



## User Header

Contains information about the record dictionary, format. User specified parameters related to conditions of the experiment.

## Data Record

Compressed buffer of data consisting of events and index. Record header provides number of events and the TAG for the record. (Data records are typically ~8 MB. Configurable to any size)

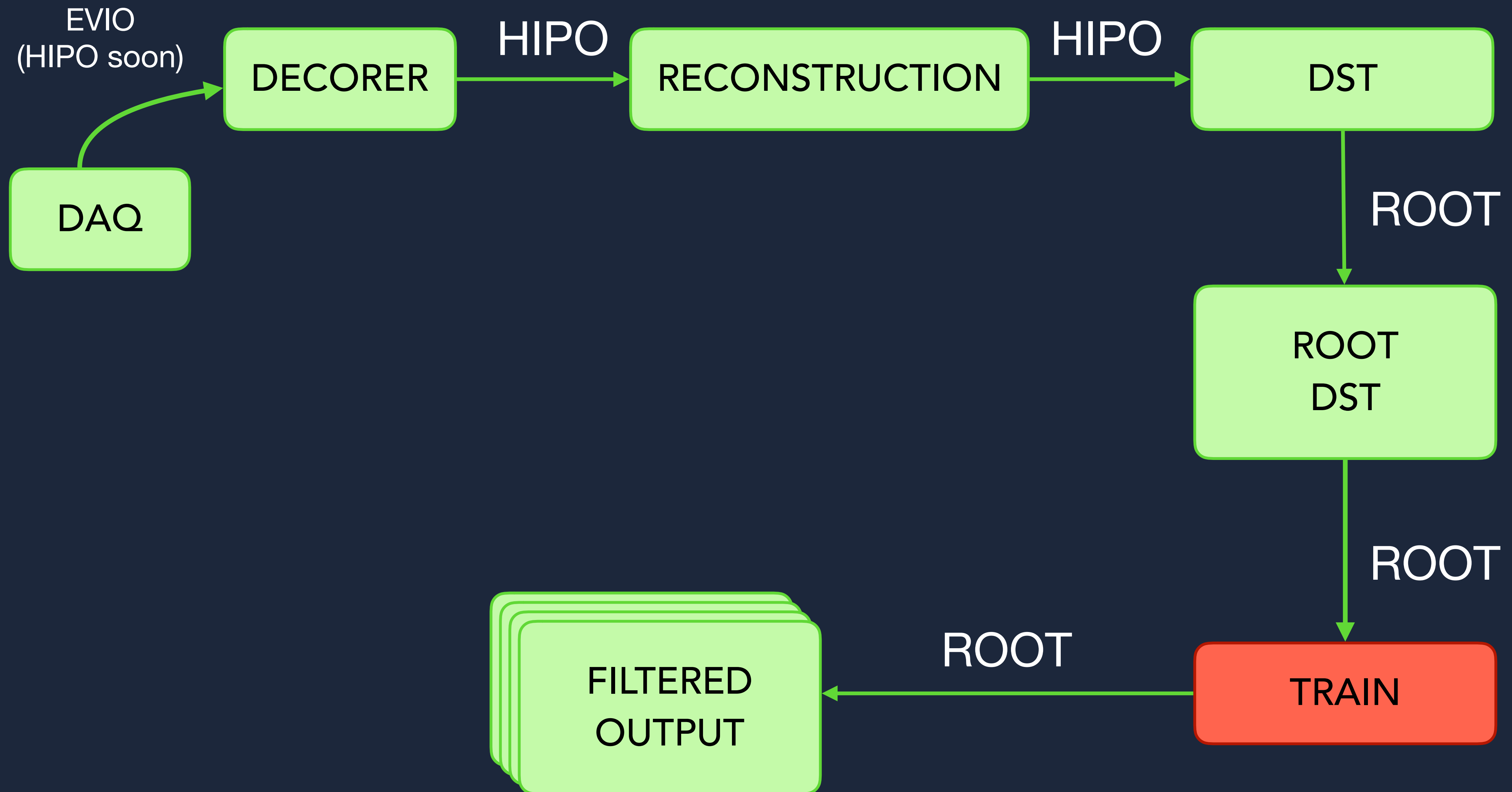
## Index Array

Array of event offsets inside the event buffer. Dynamically creates event random access table.

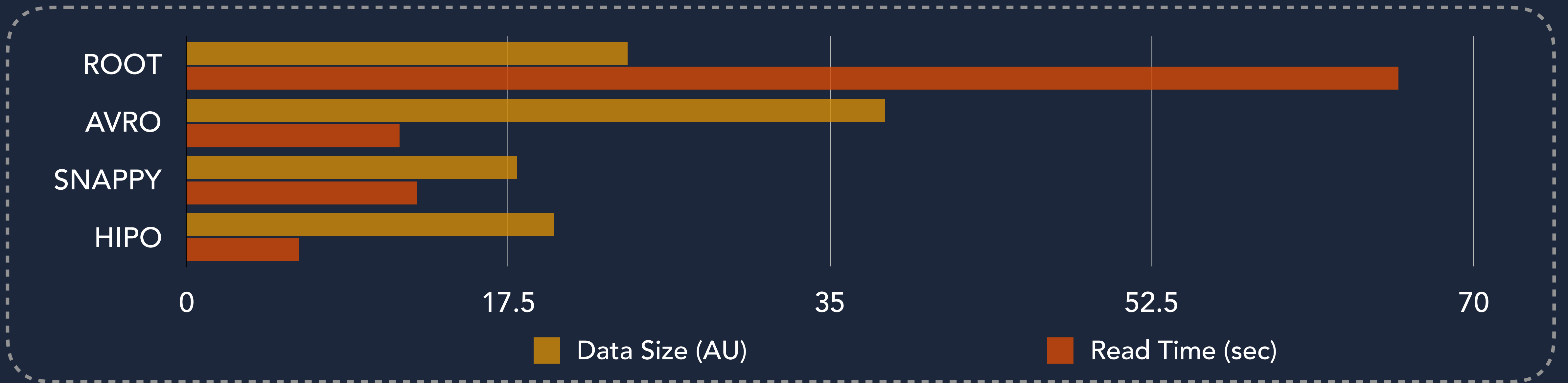
## FILE FOOTER

Contains positions of every record in the file with number of events for fast random access. Also has tags for each Data Record.

# CLAS 12 Data Flow



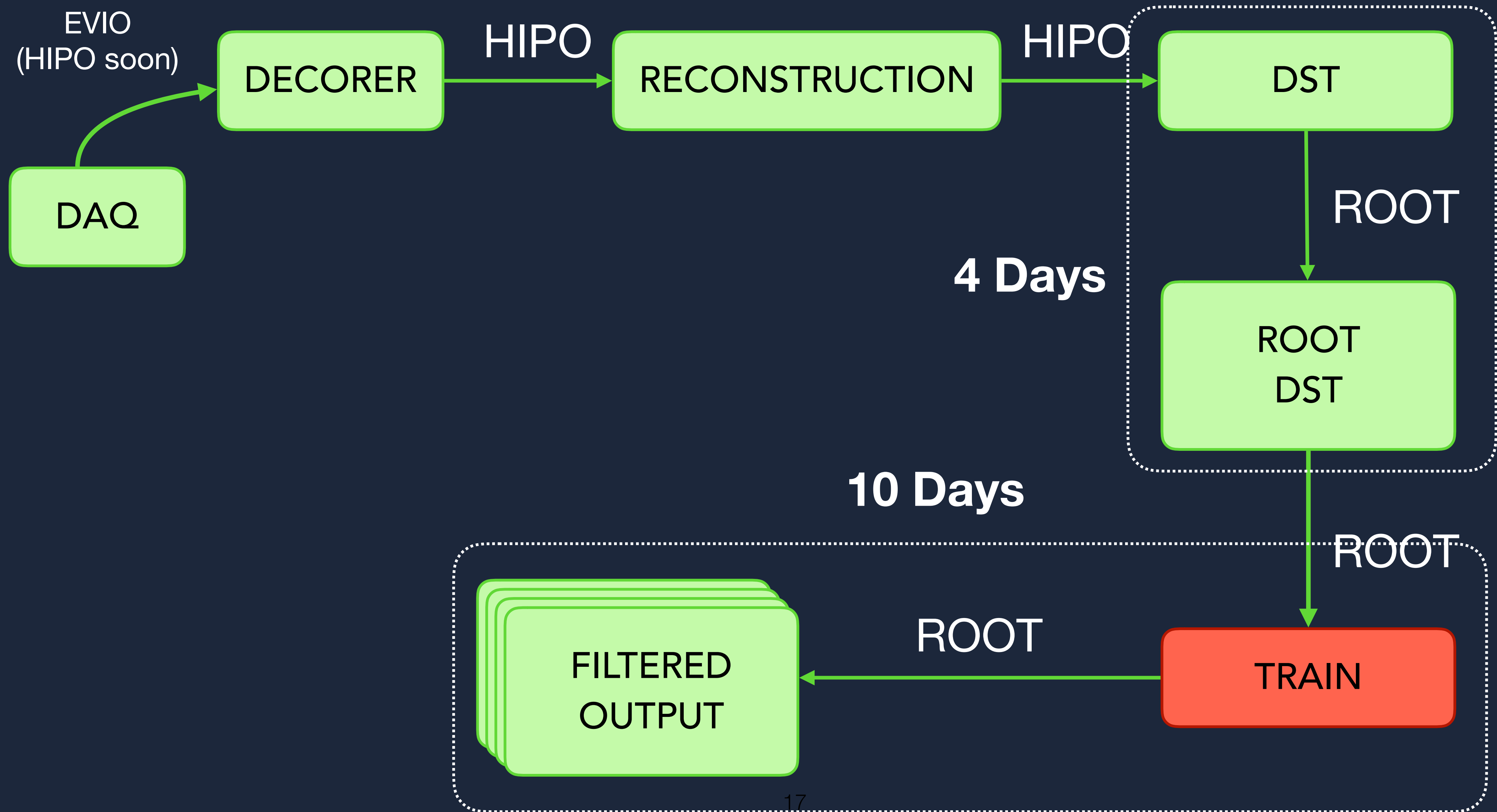
# CLAS 12 Detector



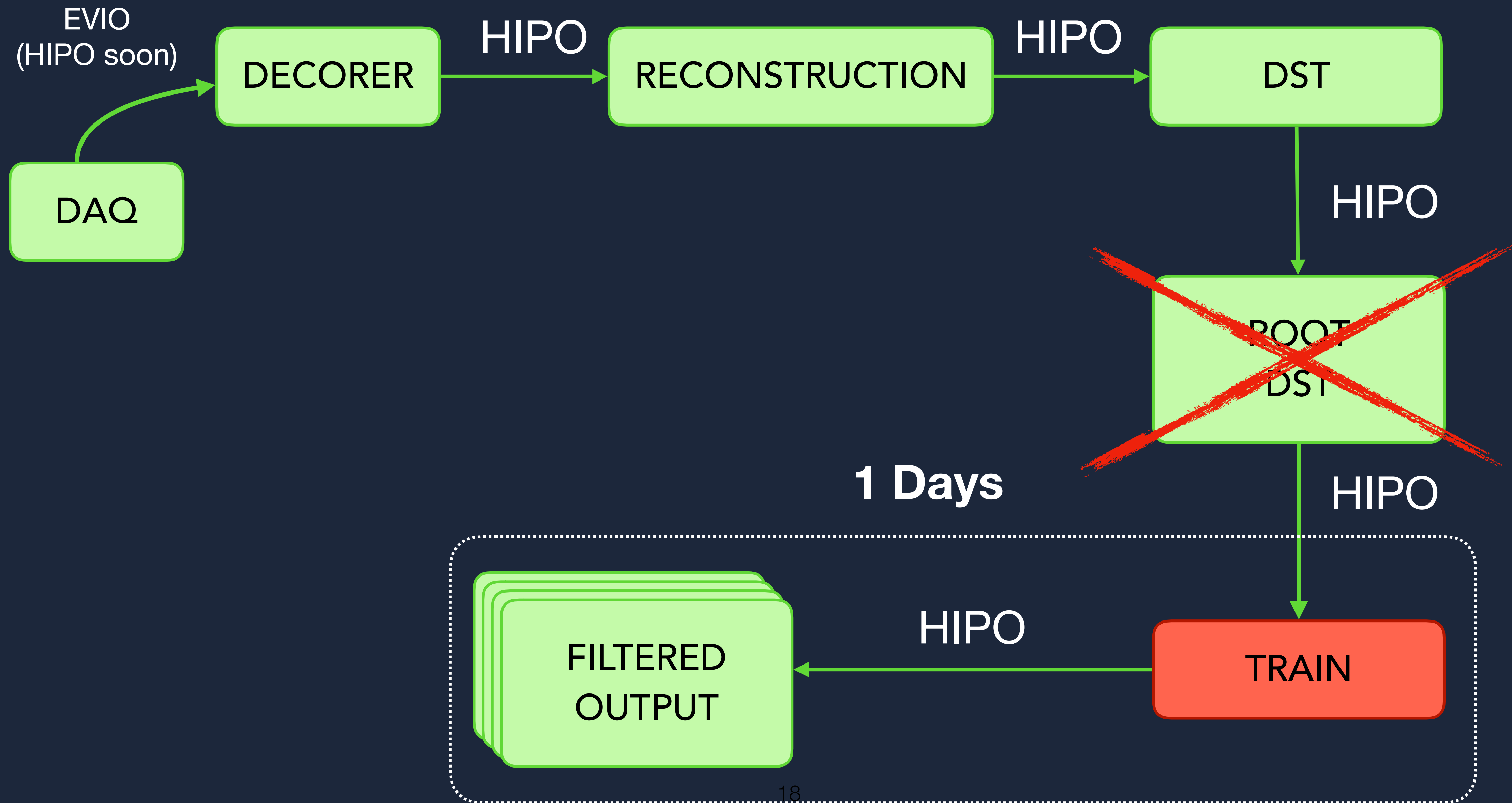
## HIGH PERFORMANCE OUTPUT (HIPO):

- HIPO data format was implemented for CLAS12
- Dictionary driven data format with Schema evolution.
- Fast compression algorithm (LZ4)
- Chunked data frame implementation to support multithreaded applications.
- Event chunk tagging and indexing to allow reading selective types of events from Data Summary Tapes.
- Deserialization performance better than most commonly used data formats in Nuclear Physics.
- Faster than industry standard data types.

# CLAS 12 Data Flow



# CLAS 12 Data Flow



# Data Formats

## CONVENTIONAL APPROACH

- Conventional data formats write data event after event as they come out from reconstruction.
- Each event has different topology of final state.
- Not all events are useful for given analysis.
- User has to read entire data set to determine which events are useful for his analysis.
- Not suitable for big data.

## NEW APPROACH

- HIPO data format is record based format with full indexed file structure.
- Records are tagged according user given criteria.
- Reading specific tags from data file is possible.
- Filtering data does not require parsing events in the records, so each skimming operation is on the level of disk IO.

### EVENT TOPOLOGY FROM CLAS12 (RUN #3856)



**59.7%** Trigger particle is not an electron.  
No electron Forward Tagger.

**25.6%** Electron trigger.  
Forward Detector

**14.7%** Forward Tagger  
No Electron in ECAL

# Conclusion

## PARRALEL DATA PROCESSING

- CLARA modernized the reconstruction software, provided tools for parallelization.
- The reconstruction code is now flexible, maintainable and easily debuggable. Users can develop their own versions of reconstruction code and run along with entire software.
- User analysis codes such as filtering events by event topology and kinematic fitting are also done using CLARA service oriented approach.

## DATA FORMATS

- HIPO data format improved the data storage for CLAS12, providing flexibility to have large files.
- Indexing and tagging mechanism in HIPO allows users to read only events of interest, without putting strain on file system, and made skimming into separate files thing of the past.

## ANSWERS:

- Computing development took a different route towards parallelization.
- We need more modern approaches to software architecture.
- With increasing data sizes of nuclear physics experiments, we have to re-think data storage and formats.

# BACKUP SLIDES