CLARA's adaptive workflow management system

Reactive micro-services based data processing orchestration.

V. Gyurjyan

gurjyan@jlab.org

Will address

- Heterogeneous data-processing optimization with CLARA's adaptive workload orchestration

- NUMA-aware workflow management system

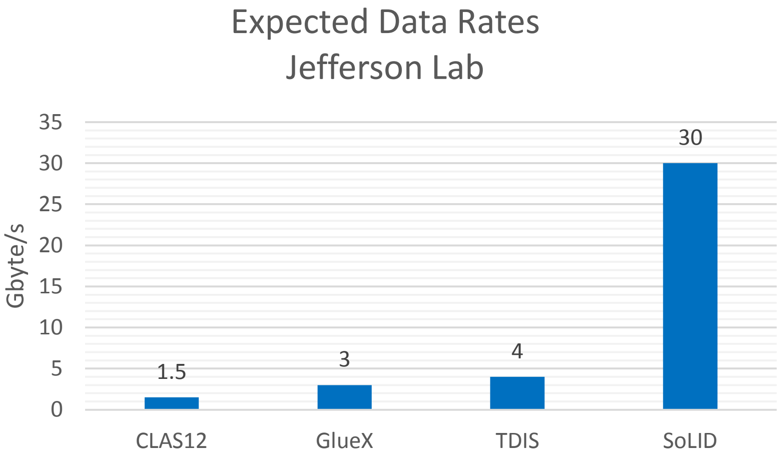
Outline

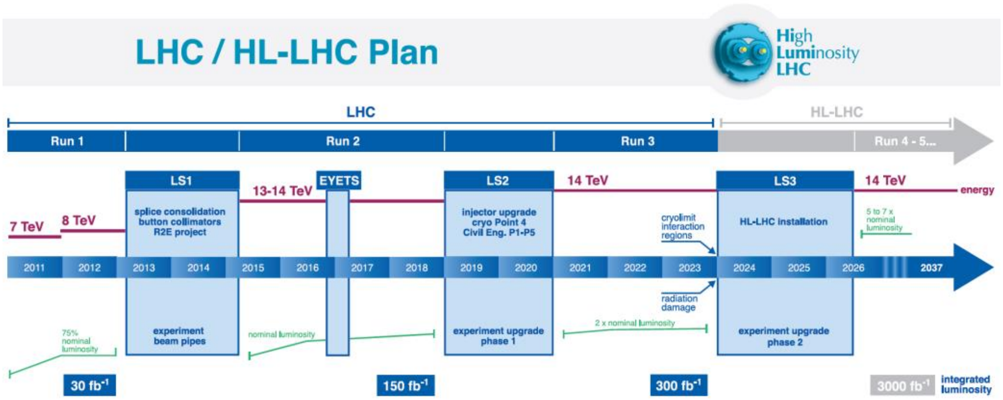
- Problem statement
- Micro-services vs Monolithic architecture
- Flow-based programming paradigm
  - Passive vs Reactive programming
  - Event vs message driven communication
- CLARA reactive micro-services based data-stream processing framework.
- Framework level workflow orchestration
- Data-processing performance optimization across diverse hardware and software infrastructures.

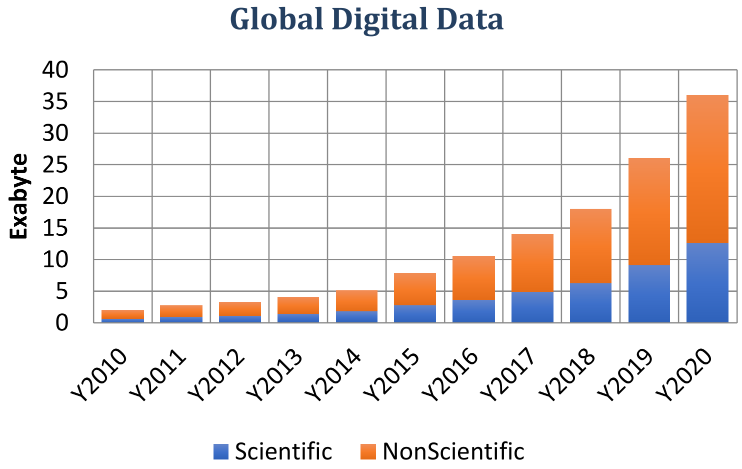
JLAB CLAS12

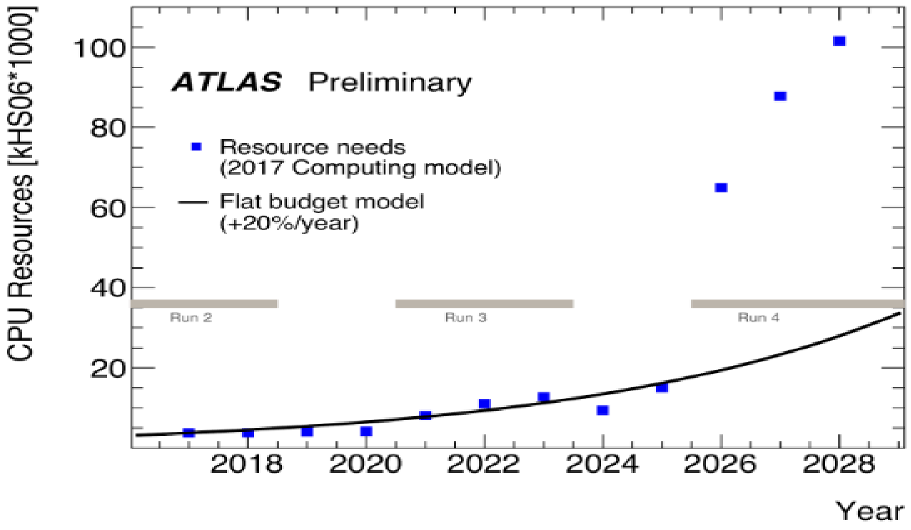
- Thomas Jefferson National Accelerator Facility (TJNAF), commonly known as Jefferson Lab or JLab, is a U.S. national laboratory located in Newport News, Virginia
- Superconducting RF technology based accelerator provides 12 GeV continuous electron beam with a bunch length of less than 1 picosecond.
- Nuclear physics experiments in 4 end- stations (A,B,C,D)
- CLAS12 is a large acceptance spectrometer installed in Hall B to study
  - Quark-gluon interactions with nuclei
  - Nucleon-nucleon correlations
  - Nucleon quark structure imaging,
  - etc.

Problem we face

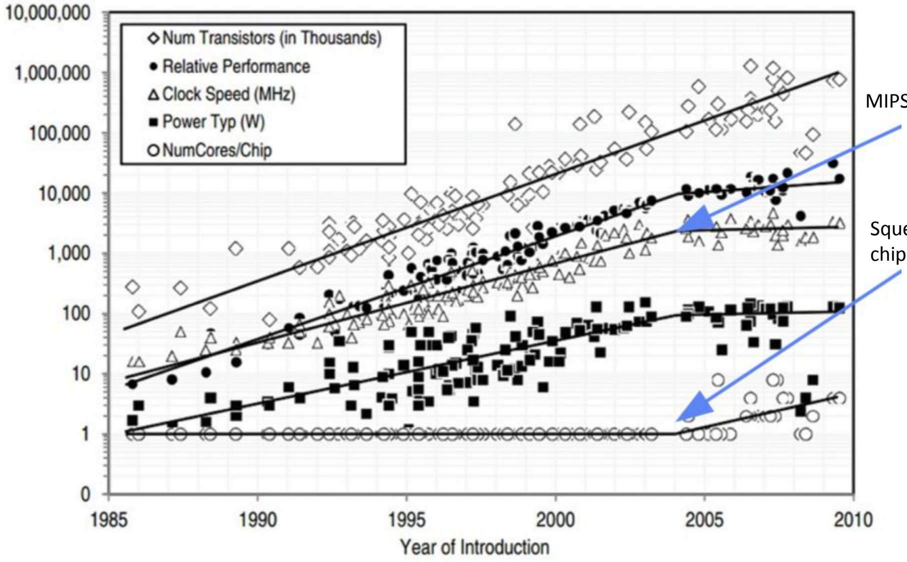








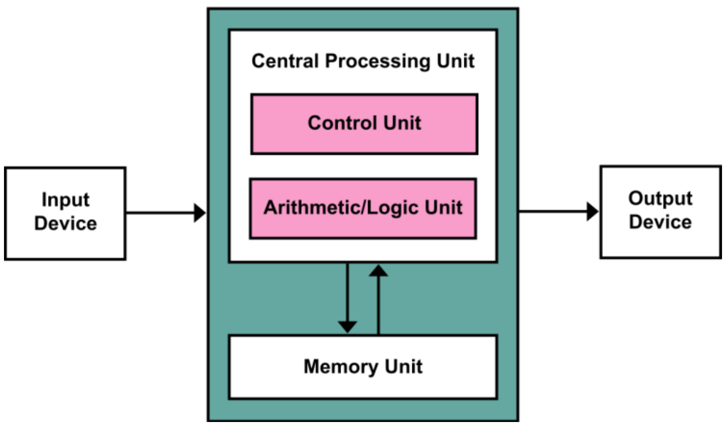
CPU based architecture limitations



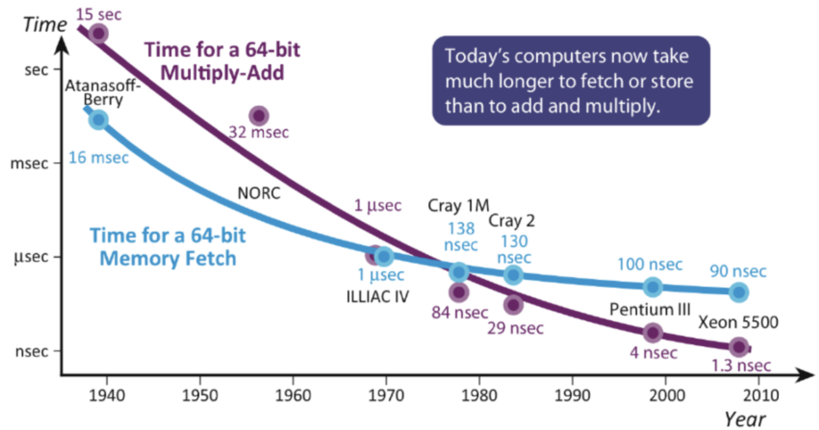
MIPS/clock speed plateau

Squeezing more cores per chip becomes difficult

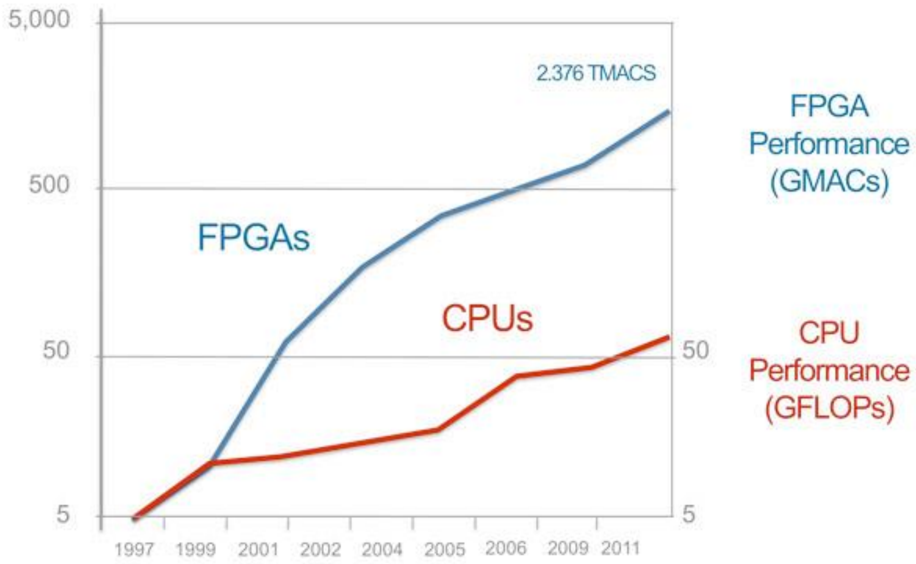
von Neumann Bottleneck



Memory Latency



Today's computers now take much longer to fetch or store than to add and multiply.

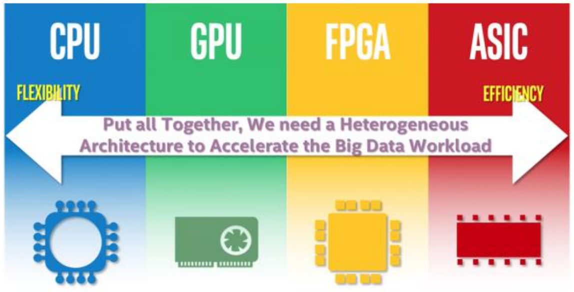


Only CPU based parallelism is not enough

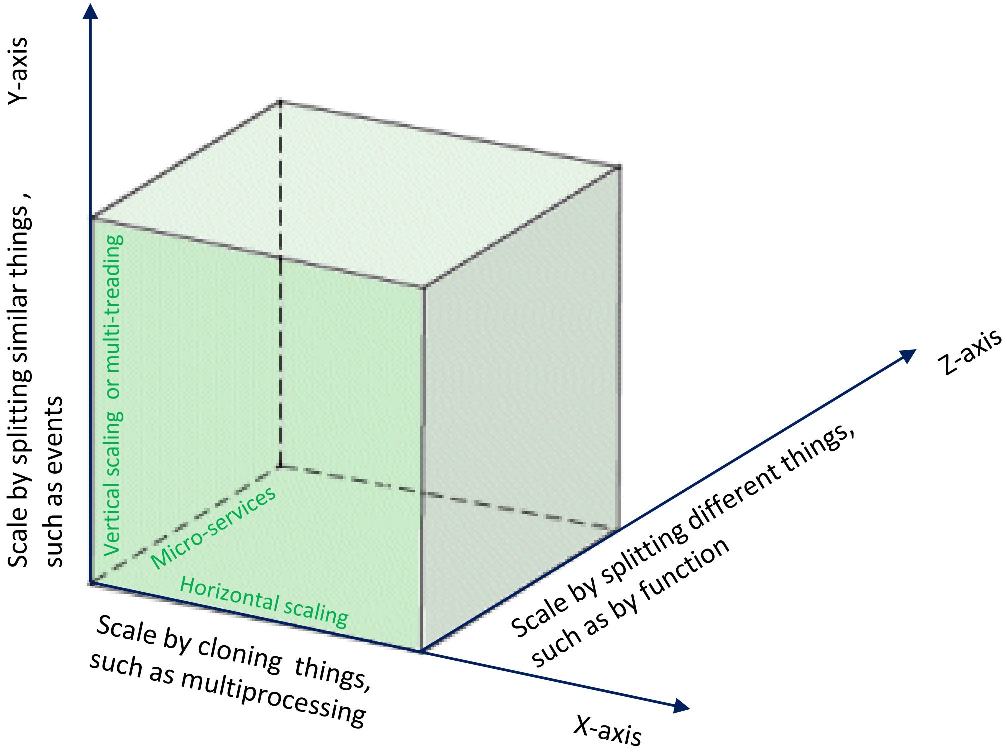
"Frameworks face the challenge of handling the massive parallelism and heterogeneity that will be present in future computing facilities, including multi-core and many-core systems, GPUs, Tensor Processing Units (TPUs), and tiered memory systems, each integrated with storage and high-speed network interconnections."

"Enable full offline analysis chains to be ported into real-time, and develop frameworks that allow non-expert offline analysis to design and deploy physics data processing systems."

A Roadmap for HEP Software and Computing R&D for the 2020s. HEP Software Foundation, Feb. 2018

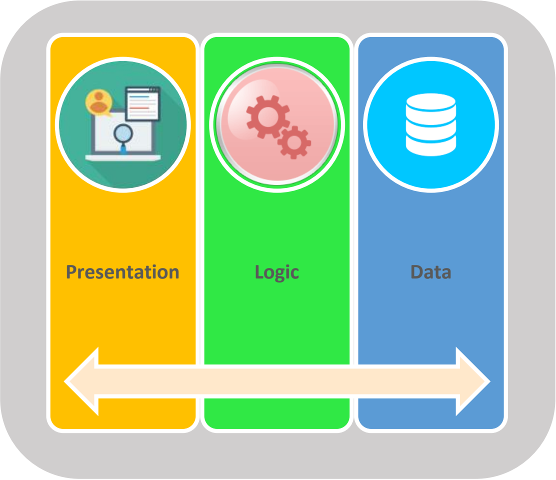

CPU
GPU
FPGA
ASIC
FLEXIBILITY
EFFICIENCY
Put all Together, We need a Heterogeneous Architecture to Accelerate the Big Data Workload

The Scale-Cube



The Art of Scalability. by Martin L. Abbott and Michael T. Fisher. **ISBN-13:** 978-0134032801

Micro-services vs Monolithic architecture

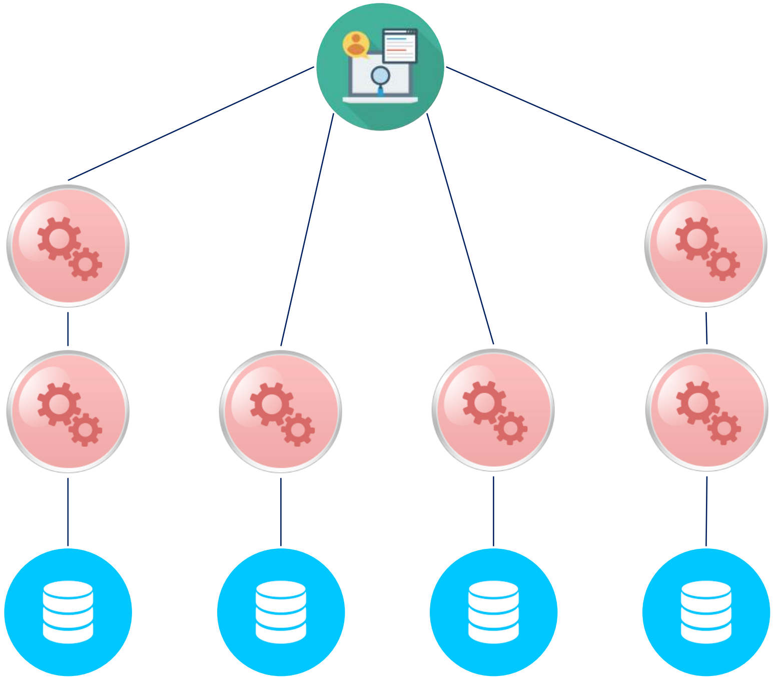


Pros

- Strong coupling, thus better performance
- Full control of your application

Cons

- No agility for isolating, compartmentalizing and decoupling data processing functionalities, suitable to run on diverse hardware/software infrastructures
- No agility for rapid development or scalability



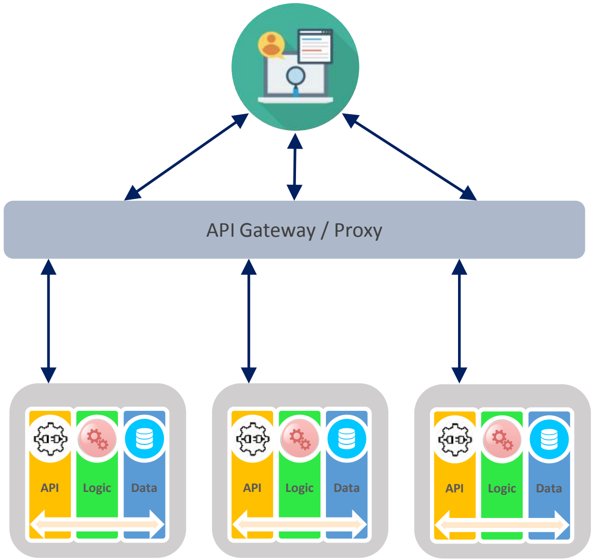
Pros

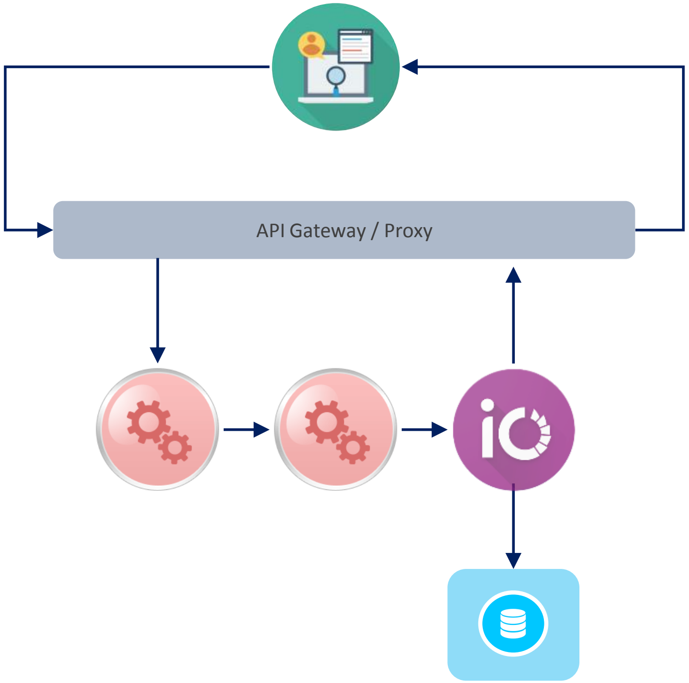
- Technology independent
- Fast iterations
- Small teams
- Fault isolation
- Scalable

Cons

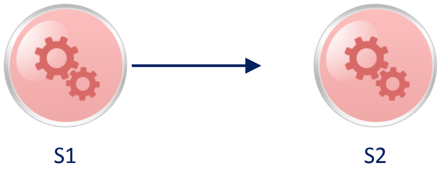
- Complexity networking (distributed system)
- Requires administration and real-time orchestration

What is micro about a service?





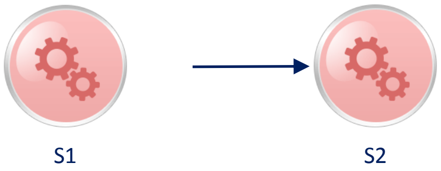
Passive vs Reactive


S1

S2

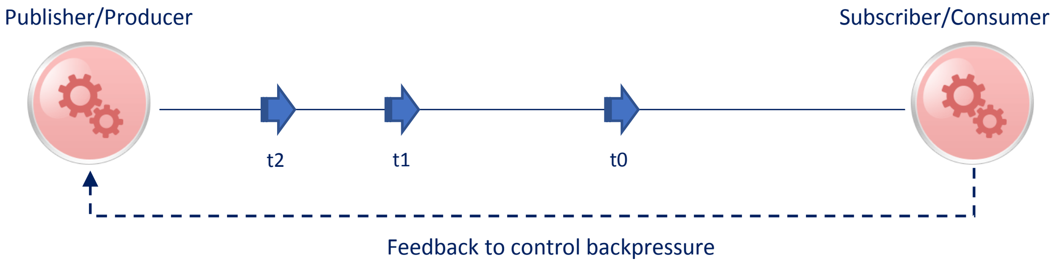
- S1: Proactive, responsible for change in S2
- S2: Passive, unaware of the dependency

Passive programming


S1

S2

- S1: Broadcasts it's own result
- S2: Subscribes S1 change events and changes itself


Publisher/Producer

Subscriber/Consumer

t2

t1

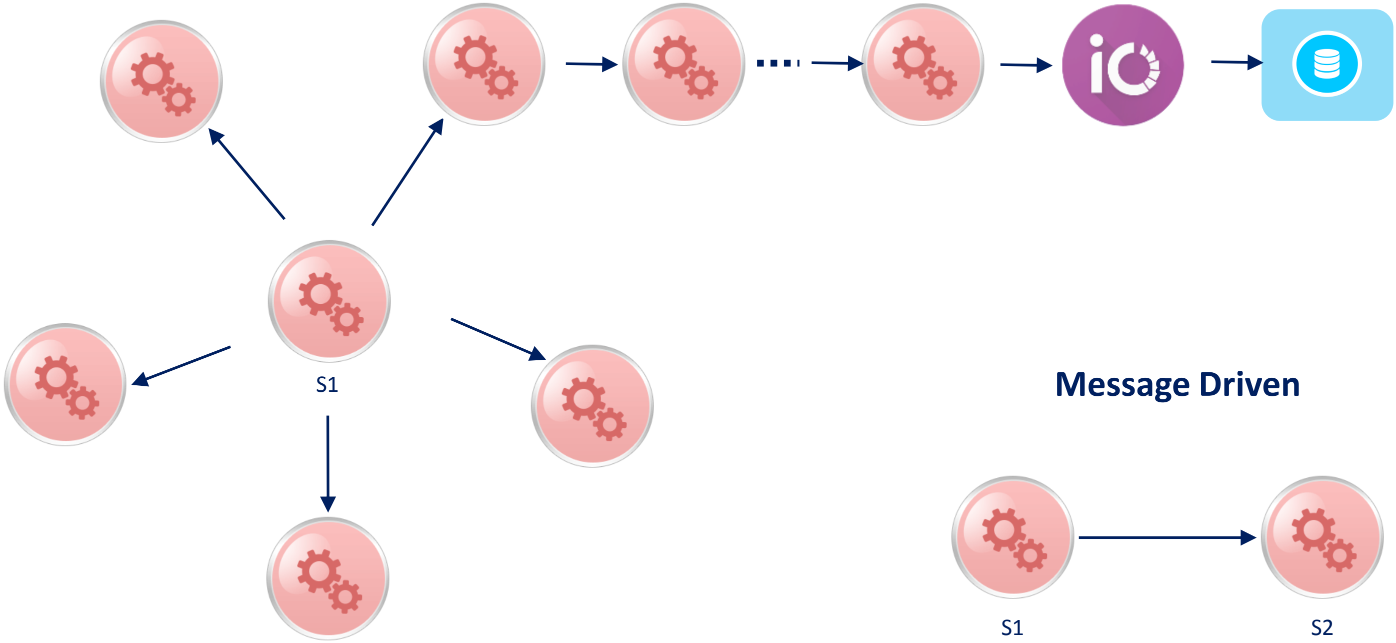
t0

Feedback to control backpressure

Reactive programming
Enables event driven stream processing

Event-Driven vs Message-Driven

Event Driven



Message Driven

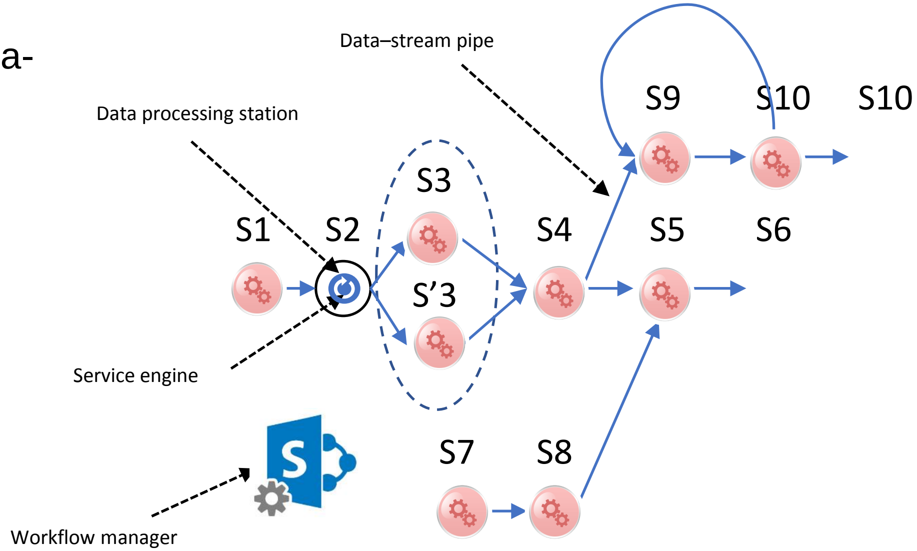
S1 event broadcasting

S1 message has a clear destination

CLARA Framework

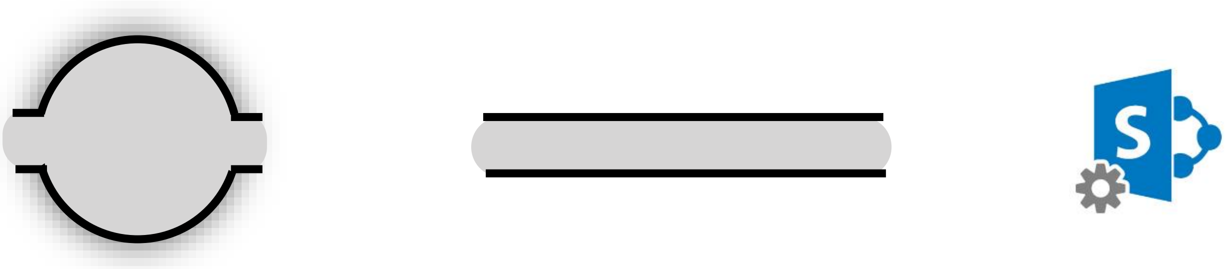
Reactive, event-driven data-stream processing framework that implements micro-services architecture and FBP

- Defines streaming transient-data structure
- Provides service abstraction (data processing station) to present user algorithm (engine) as an independent service.
- Defines service communication channel (data-stream pipe) outside of the user engine.
- Stream-unit level workflow management system and API
- Supports C++, JAVA, Python languages



http://claraweb.jlab.org
https://claraweb.jlab.org/clara/docs/clas/hands-on.html

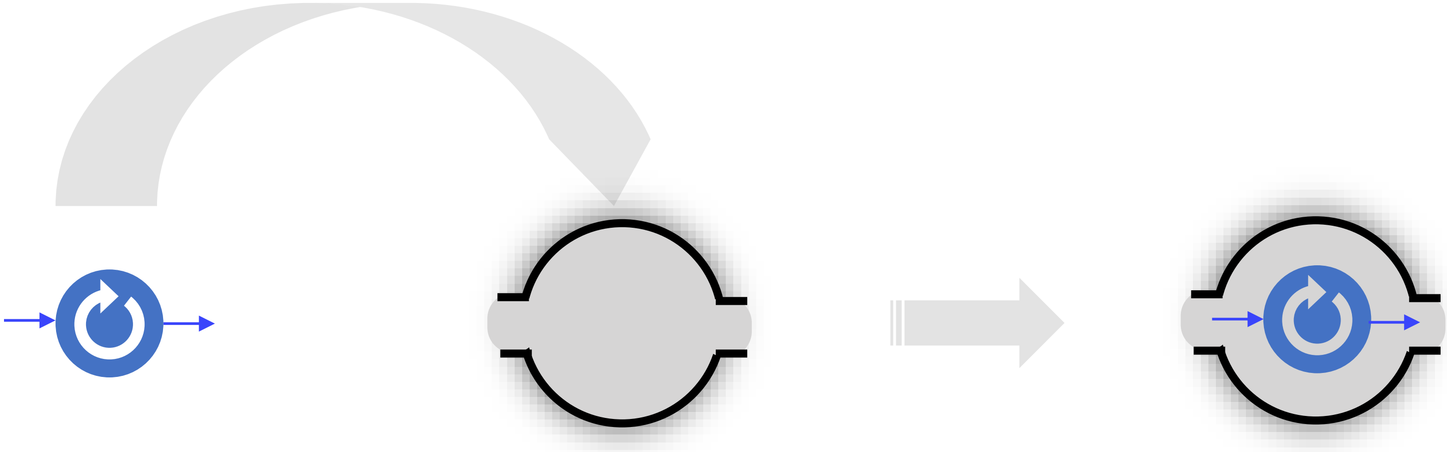
Basic components and a user code interface



Data Processing Station

Data-Stream Pipe

Orchestrator



Data processing Engine

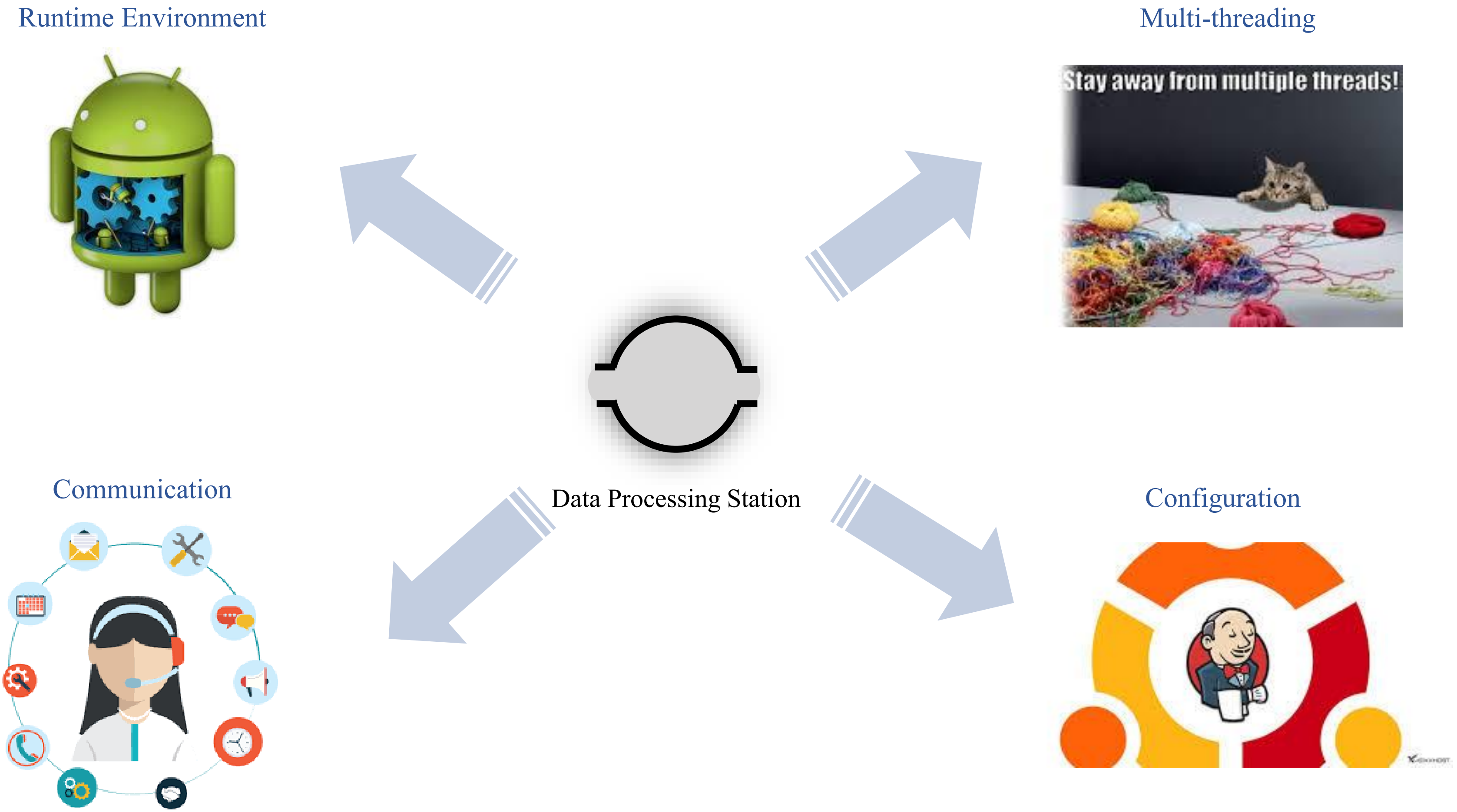
Data Processing Station

Data Processing Micro-Service

Engine Tutorials

- https://claraweb.jlab.org/clara/docs/quickstart/java.html
- https://claraweb.jlab.org/clara/docs/quickstart/cpp.html
- https://claraweb.jlab.org/clara/docs/quickstart/python.html

Data Processing Station



Runtime Environment

Multi-threading

Communication

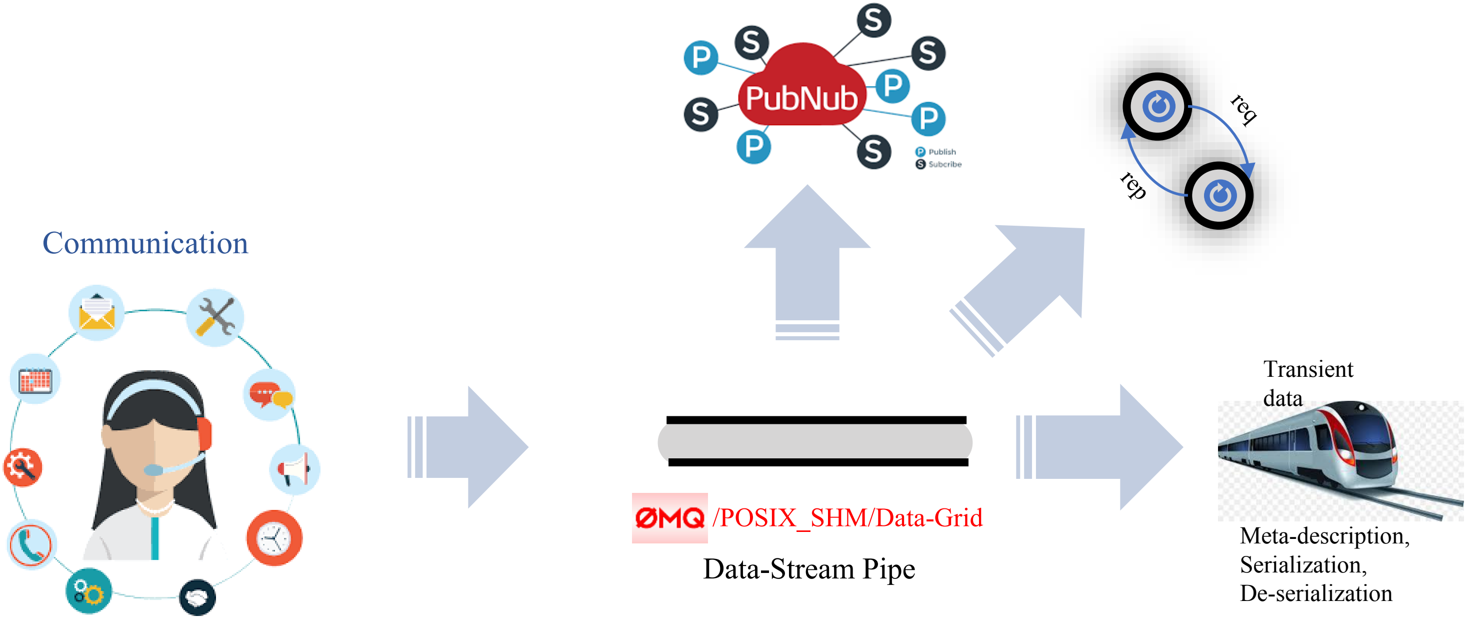
Data Processing Station

Configuration

Language Bindings

- https://github.com/JeffersonLab/clara-java.git
- https://github.com/JeffersonLab/clara-cpp.git
- https://github.com/JeffersonLab/clara-python.git

Data Stream Pipe



Communication

Transient data

/POSIX_SHM/Data-Grid

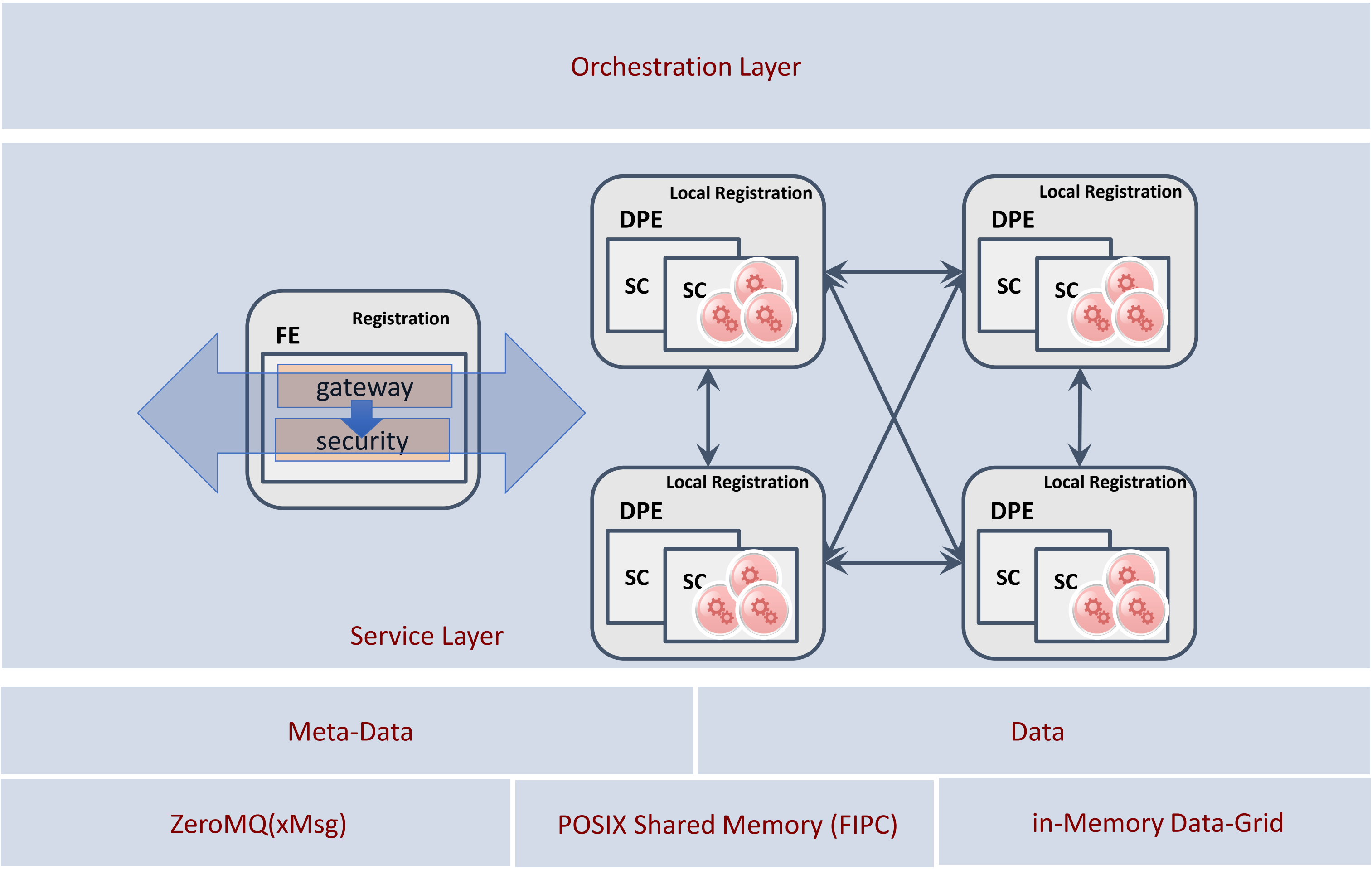
Data-Stream Pipe

Meta-description,
Serialization,
De-serialization

Language Bindings

- https://github.com/JeffersonLab/xmsg-java.git
- https://github.com/JeffersonLab/xmsg-cpp.git
- https://github.com/JeffersonLab/xmsg-python.git

Structure



Orchestration Layer

FE

Registration

gateway

security

Local Registration

DPE

SC

SC

Local Registration

DPE

SC

SC

Local Registration

DPE

SC

SC

Local Registration

DPE

SC

SC

Service Layer

Meta-Data

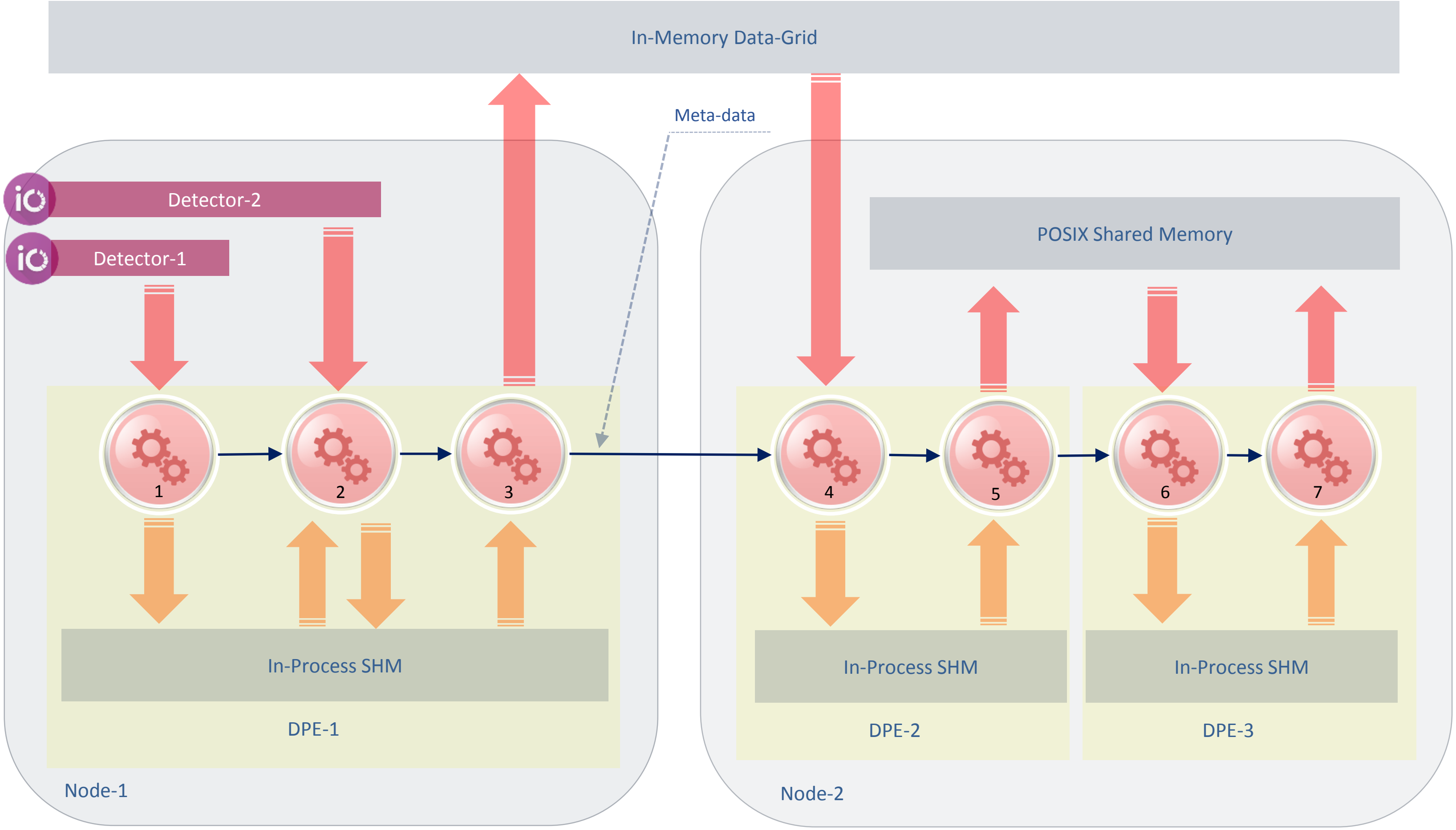
Data

ZeroMQ(xMsg)

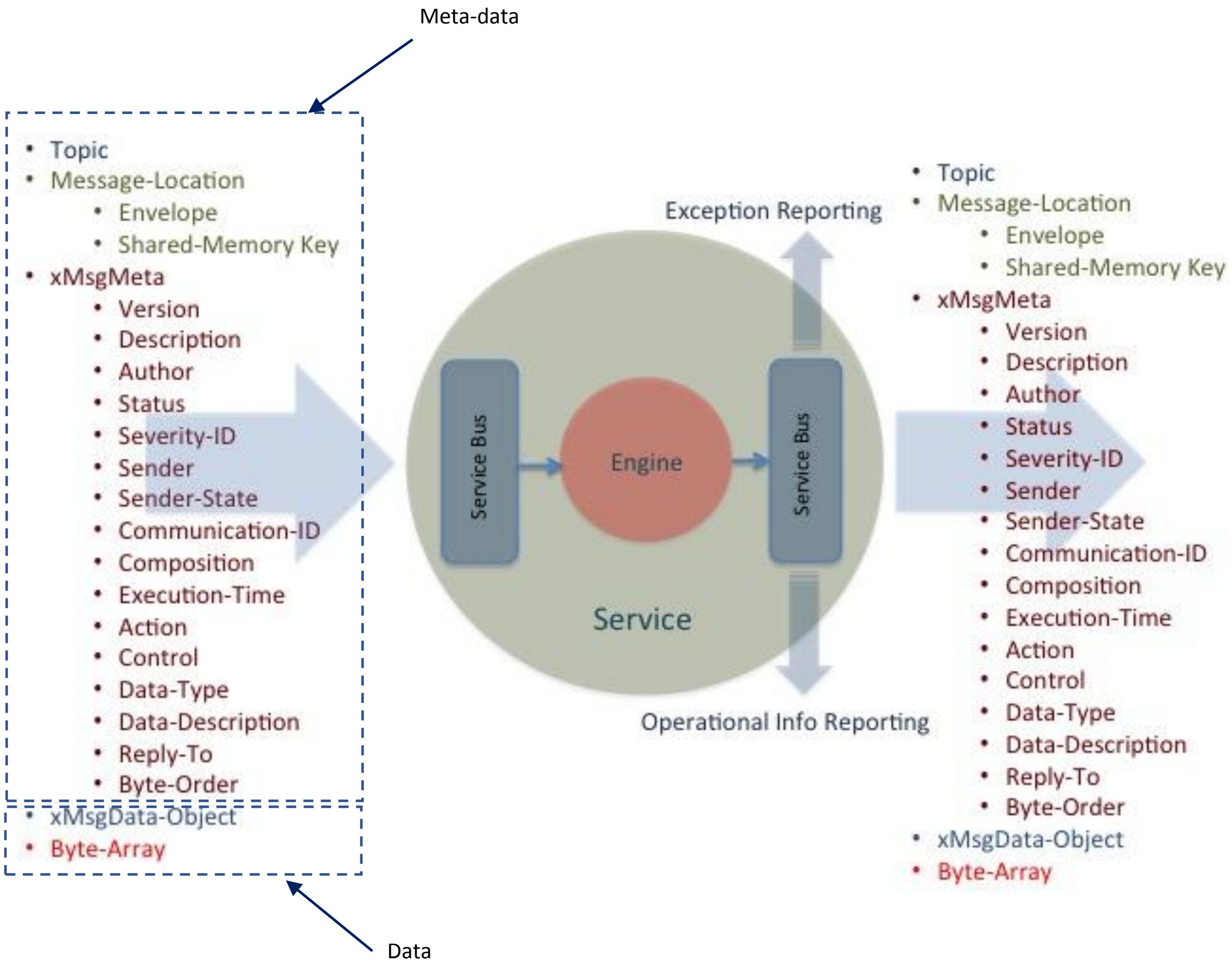
POSIX Shared Memory (FIPC)

in-Memory Data-Grid

Streaming Data-Flow Processing



Transient data unit (meta-data + data)



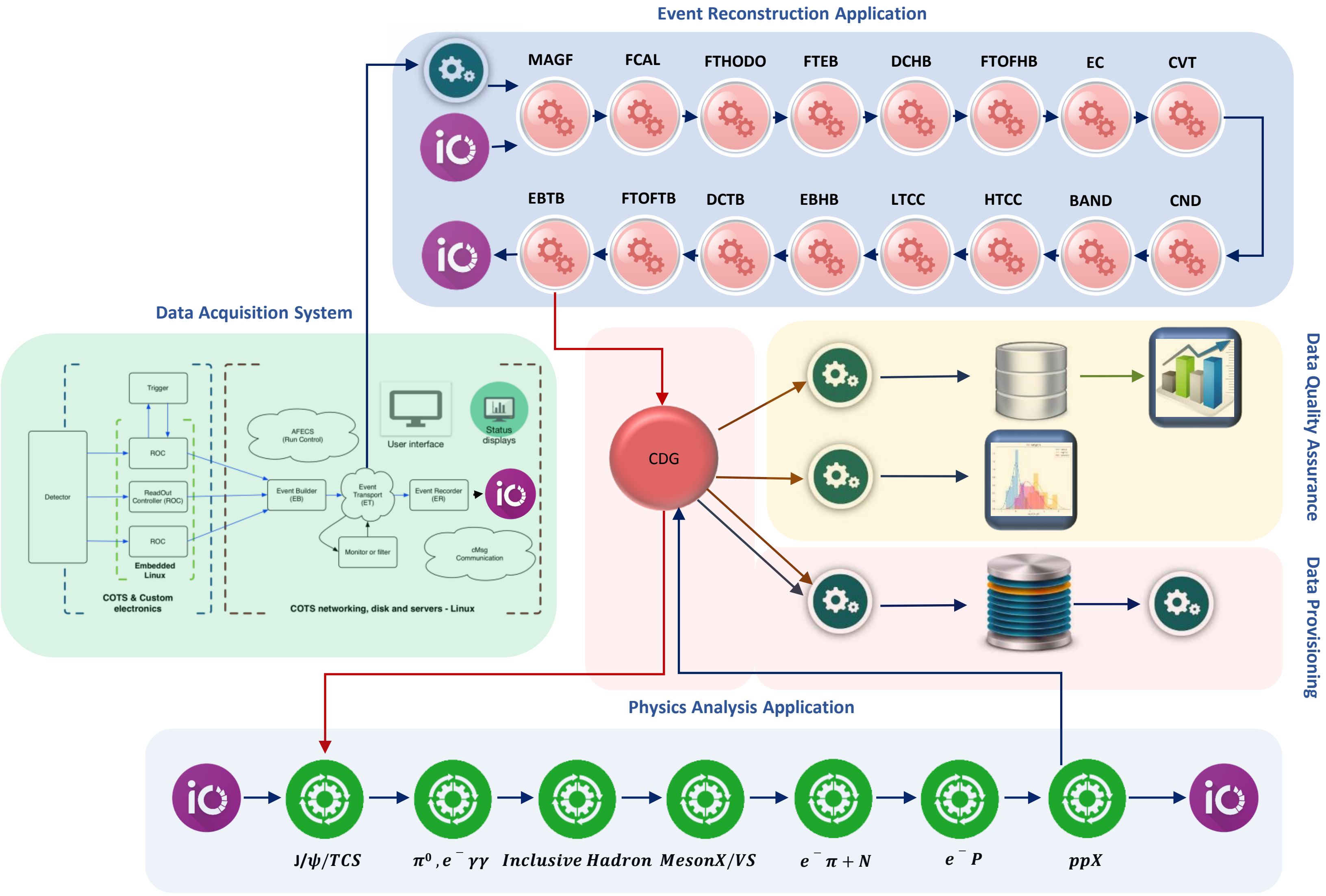
Meta-data

- Topic
- Message-Location
    - Envelope
    - Shared-Memory Key
- xMsgMeta
    - Version
    - Description
    - Author
    - Status
    - Severity-ID
    - Sender
    - Sender-State
    - Communication-ID
    - Composition
    - Execution-Time
    - Action
    - Control
    - Data-Type
    - Data-Description
    - Reply-To
    - Byte-Order
- xMsgData-Object
- Byte-Array

Data

- Topic
- Message-Location
    - Envelope
    - Shared-Memory Key
- xMsgMeta
    - Version
    - Description
    - Author
    - Status
    - Severity-ID
    - Sender
    - Sender-State
    - Communication-ID
    - Composition
    - Execution-Time
    - Action
    - Control
    - Data-Type
    - Data-Description
    - Reply-To
    - Byte-Order
- xMsgData-Object
- Byte-Array

CLAS12 Data Processing Applications



Event Reconstruction Application

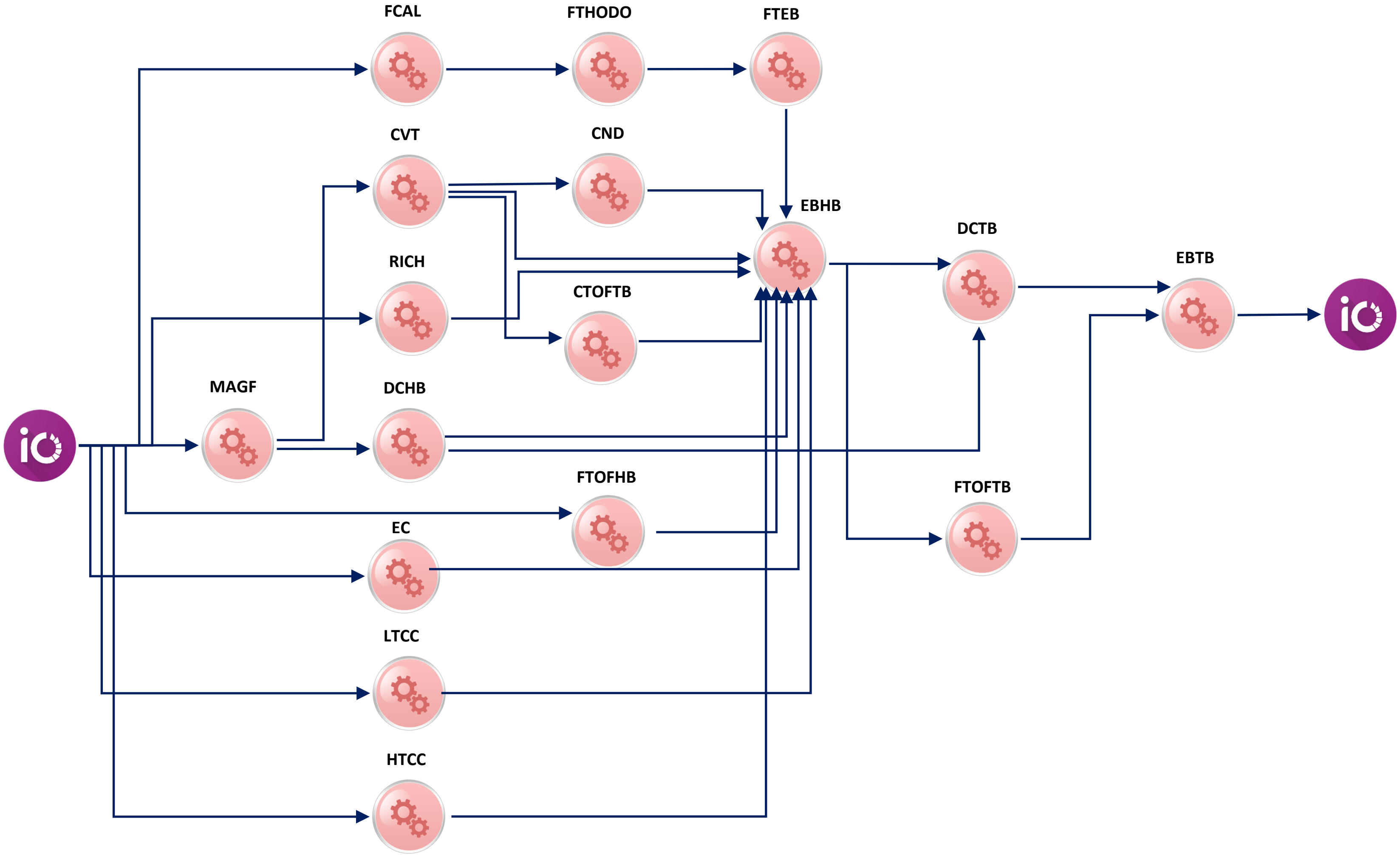
Data Acquisition System

Data Quality Assurance

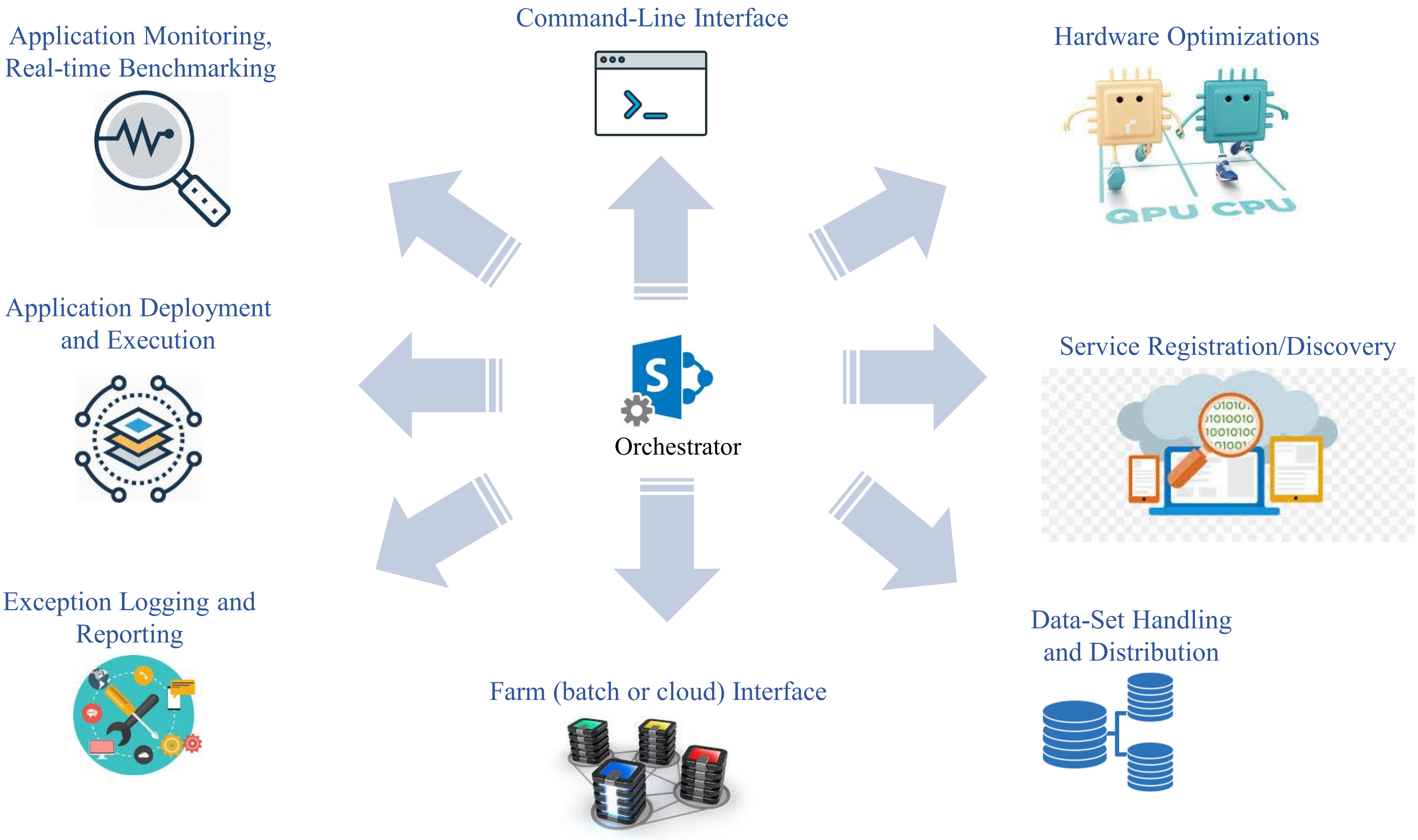
Data Provisioning

Physics Analysis Application

Event Reconstruction Application (sub-event level parallelization)



Workflow orchestrator



Application Monitoring, Real-time Benchmarking

Command-Line Interface

Hardware Optimizations

Application Deployment and Execution

Orchestrator

Service Registration/Discovery

Exception Logging and Reporting

Farm (batch or cloud) Interface

Data-Set Handling and Distribution

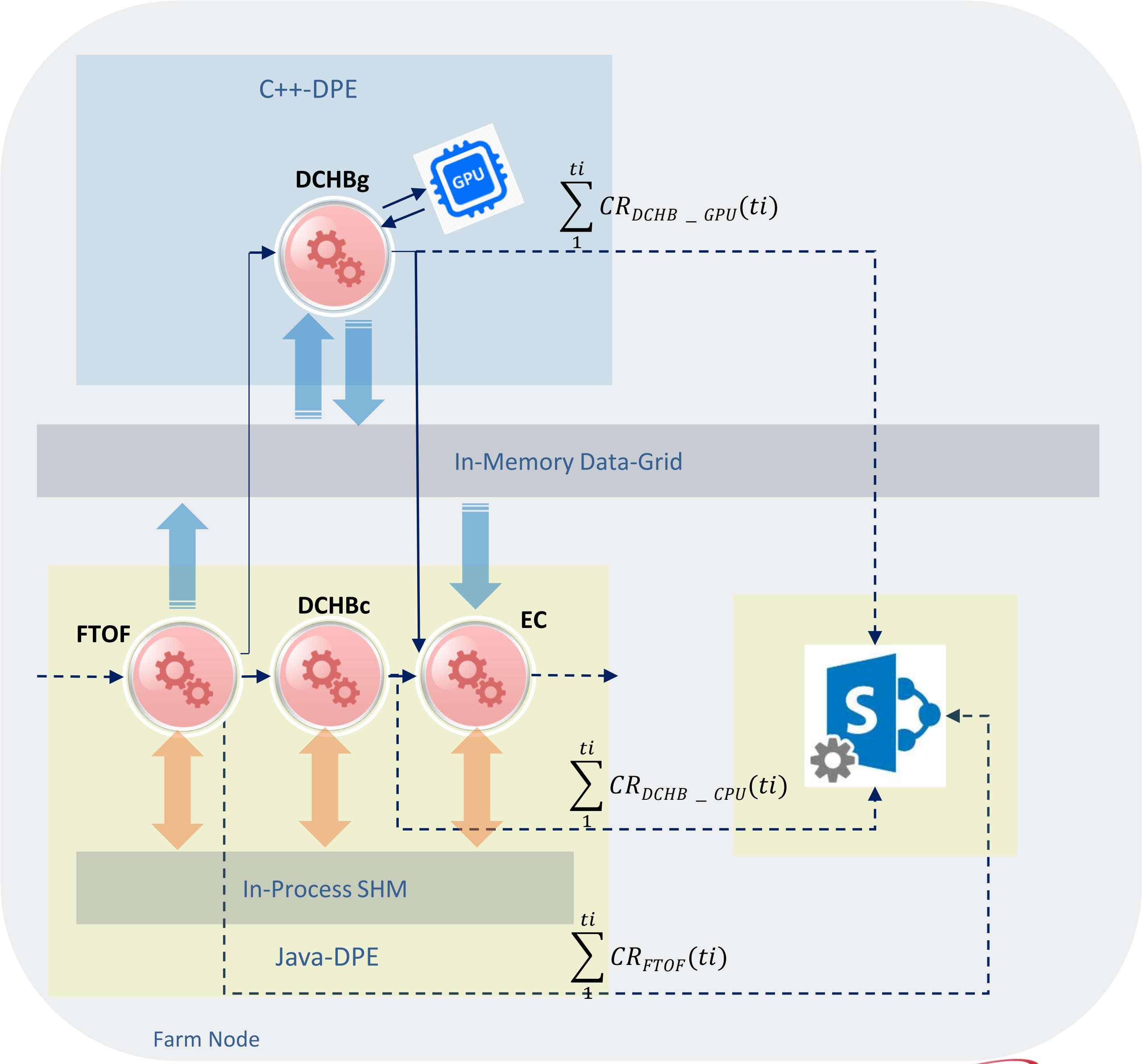
Heterogeneous deployment algorithm

$$P_g = \frac{\sum_1^{ti} CR_{FTOF}(ti)}{\sum_1^{ti} CR_{DCHB\_GPU}(ti)}$$

$$P_c = \frac{\sum_1^{ti} CR_{FTOF}(ti)}{\sum_1^{ti} CR_{DCHB\_CPU}(ti)}$$

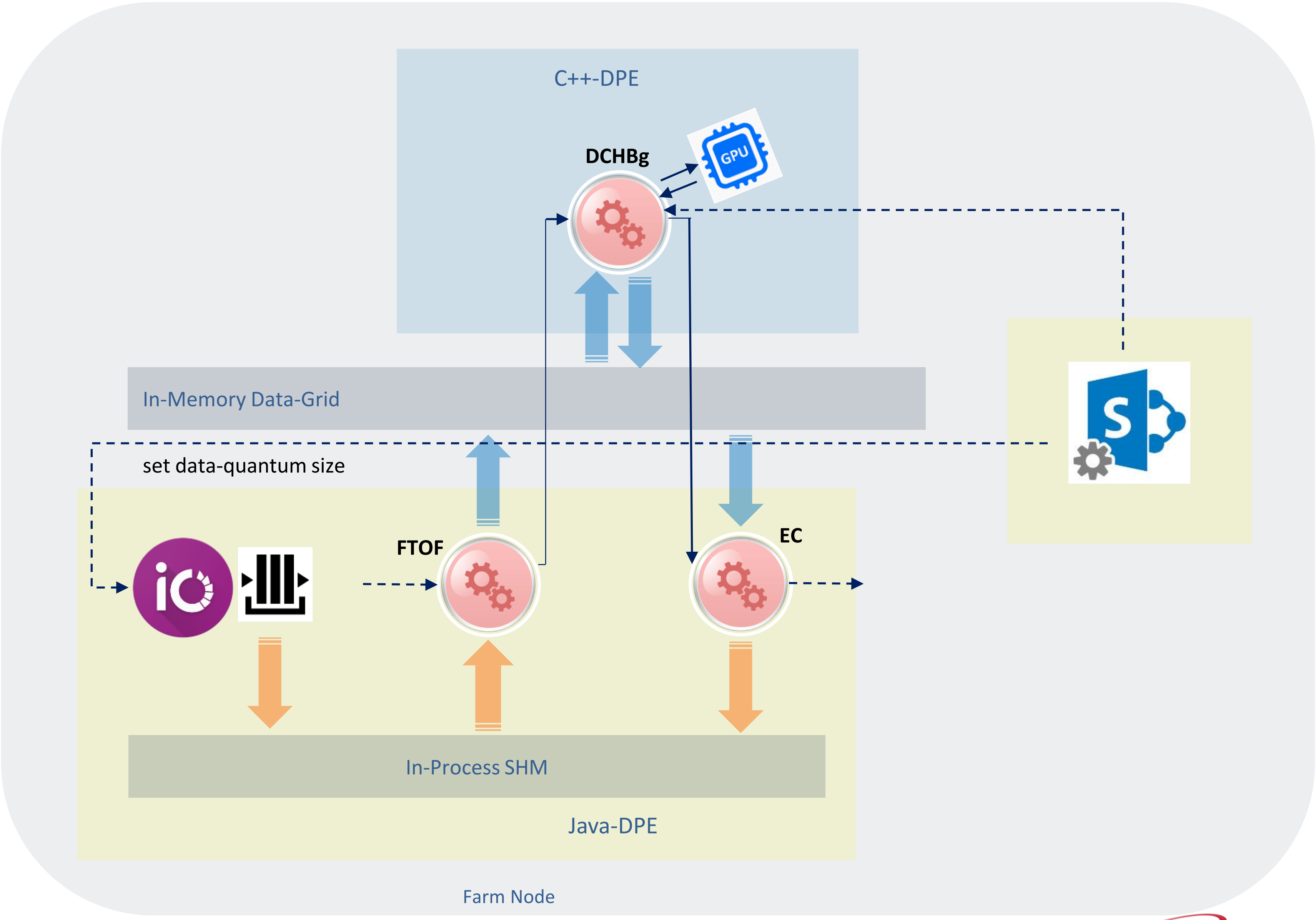
if $Pg < Pc$

route data–stream through DCHBg



C++-DPE
DCHBg
GPU
$\sum_1^{ti} CR_{DCHB\_GPU}(ti)$
In-Memory Data-Grid
FTOF
DCHBc
EC
$\sum_1^{ti} CR_{DCHB\_CPU}(ti)$
In-Process SHM
Java-DPE
$\sum_1^{ti} CR_{FTOF}(ti)$
Farm Node

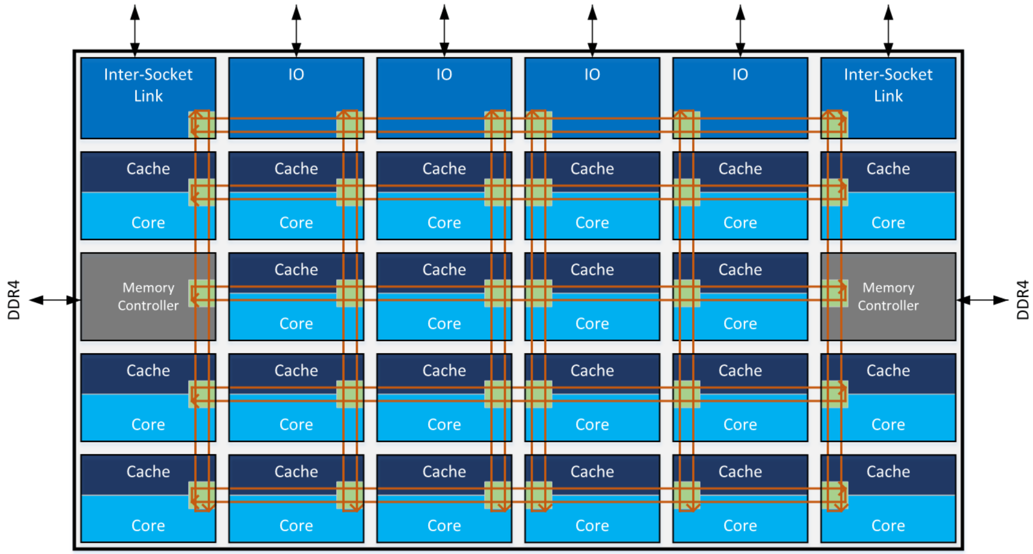
Data-quantum size and GPU occupancy



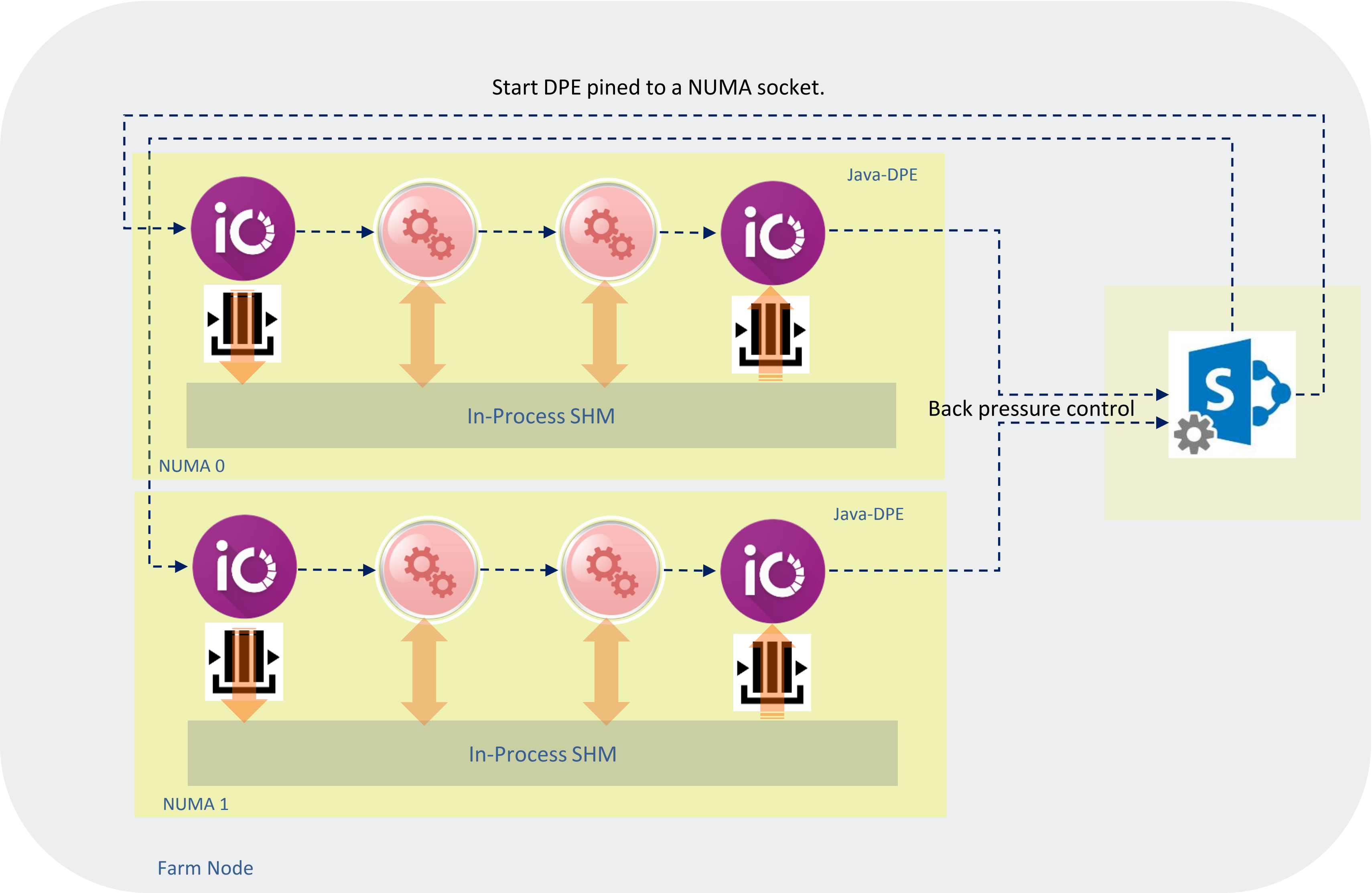
Thread motion and DVFS

- Per-core, independent voltage control becomes impractical
- Limited number of independent DVFS systems for multicore systems
- Large core density systems are deploying a new power management technique that migrates threads from core to core to adjust power and performance to the time-varying needs of a running program.

Intel Mesh Interconnect Architecture

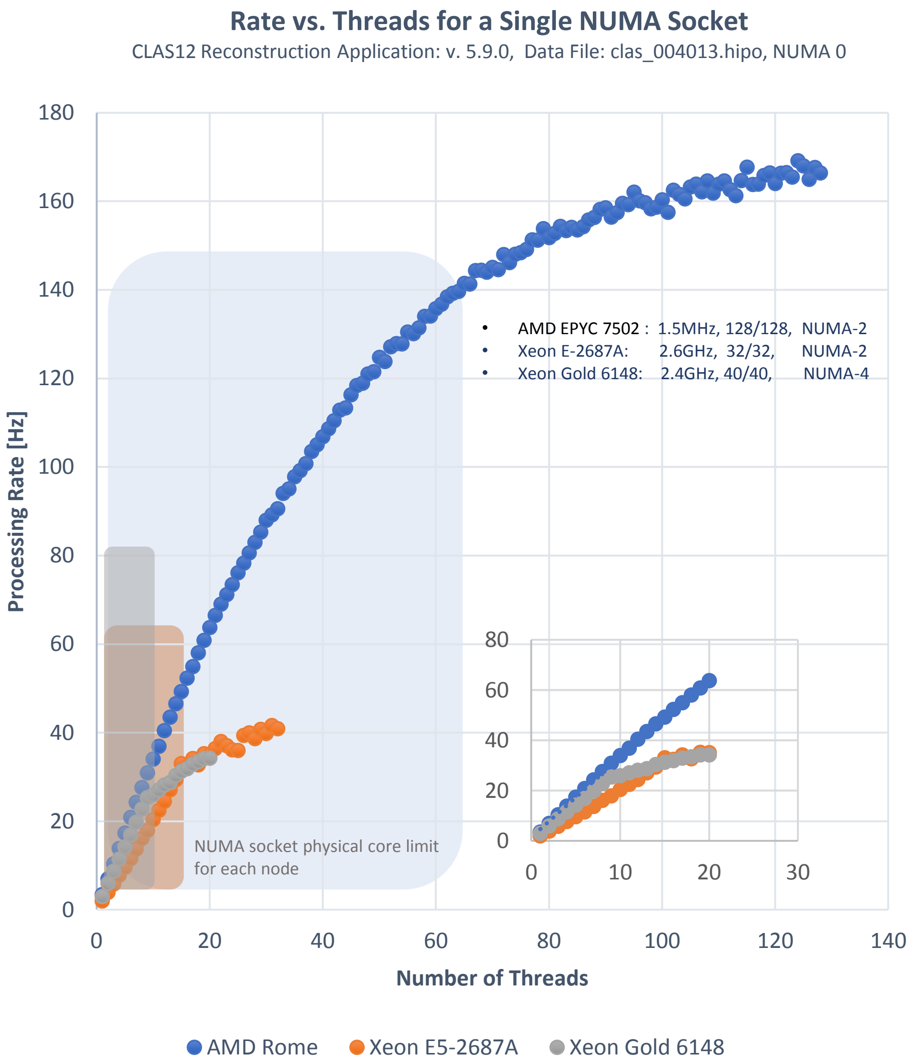


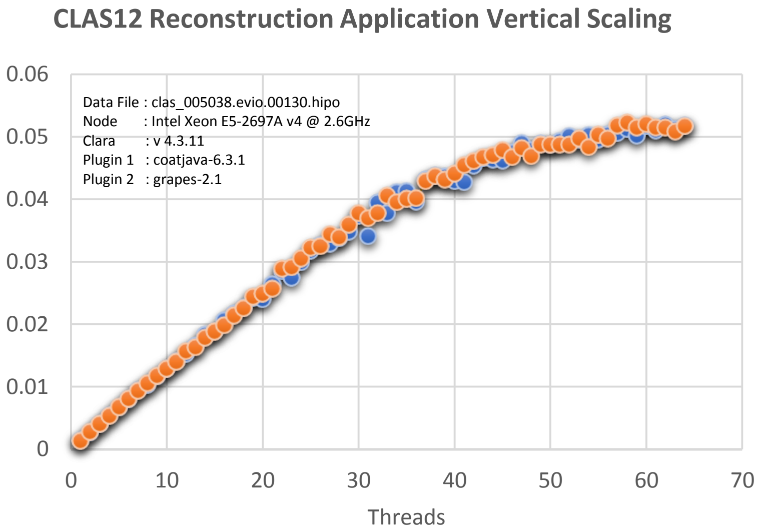
Data-processing chain per NUMA

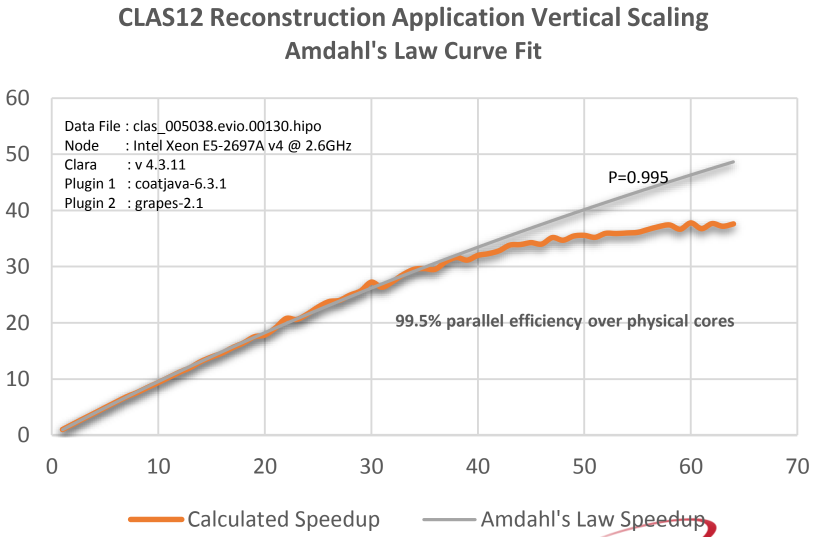


Start DPE pined to a NUMA socket.

Results


Rate vs. Threads for a Single NUMA Socket
CLAS12 Reconstruction Application: v. 5.9.0, Data File: clas_004013.hipo, NUMA 0
Processing Rate [Hz]
180
160
140
120
100
80
60
40
20
0
0
20
40
60
80
100
120
140
Number of Threads
• AMD EPYC 7502 : 1.5MHz, 128/128, NUMA-2
• Xeon E-2687A: 2.6GHz, 32/32, NUMA-2
• Xeon Gold 6148: 2.4GHz, 40/40, NUMA-4
NUMA socket physical core limit for each node
80
60
40
20
0
0
10
20
30
AMD Rome
Xeon E5-2687A
Xeon Gold 6148


CLAS12 Reconstruction Application Vertical Scaling
0.06
0.05
0.04
0.03
0.02
0.01
0
0
10
20
30
40
50
60
70
Threads
Data File : clas_005038.evio.00130.hipo
Node : Intel Xeon E5-2697A v4 @ 2.6GHz
Clara : v 4.3.11
Plugin 1 : coatjava-6.3.1
Plugin 2 : grapes-2.1


CLAS12 Reconstruction Application Vertical Scaling
Amdahl's Law Curve Fit
60
50
40
30
20
10
0
0
10
20
30
40
50
60
70
Data File : clas_005038.evio.00130.hipo
Node : Intel Xeon E5-2697A v4 @ 2.6GHz
Clara : v 4.3.11
Plugin 1 : coatjava-6.3.1
Plugin 2 : grapes-2.1
P=0.995
99.5% parallel efficiency over physical cores
Calculated Speedup
Amdahl's Law Speedup

Summary

- Frameworks based on the micro-services architecture are in a better position to address massive parallelism and heterogeneity of current and future computing facilities.
- CLARA is a mature, micro-services based, data stream processing framework in production-use at JLAB and NASA.
- Internal, stream-unit level workflow management system is designed with adaptive functionalities that guarantees maximum data processing performance across diverse hardware and software infrastructures.