Data processing workflows for the SPD experiment

Danila Oleynik, MSCHEP2026

Some basic definitions

- **DATA** – a representation of facts or ideas in a formalized version, capable of being communicated or manipulated in some process.
- **INFORMATION –** in automatic data processing the meaning that a human assigns to data by means of the known conventions used in its representation.
- **DATA PROCESSING** – the execution of a systematic sequence of operations, performed with data, e.g. handling, merging, sorting, computing.
  - Note: Where data processing is performed in order to increase the value or significance (from a certain point of view) of the information conveyed by the data, it may be called **INFORMATION PROCESSING.**

THE TERMINOLOGY WORK OF IFIP (International Federation for Information Processing) AND ICC (International Computing Centre)
I. H. GOULD and G. C. TOOTILL

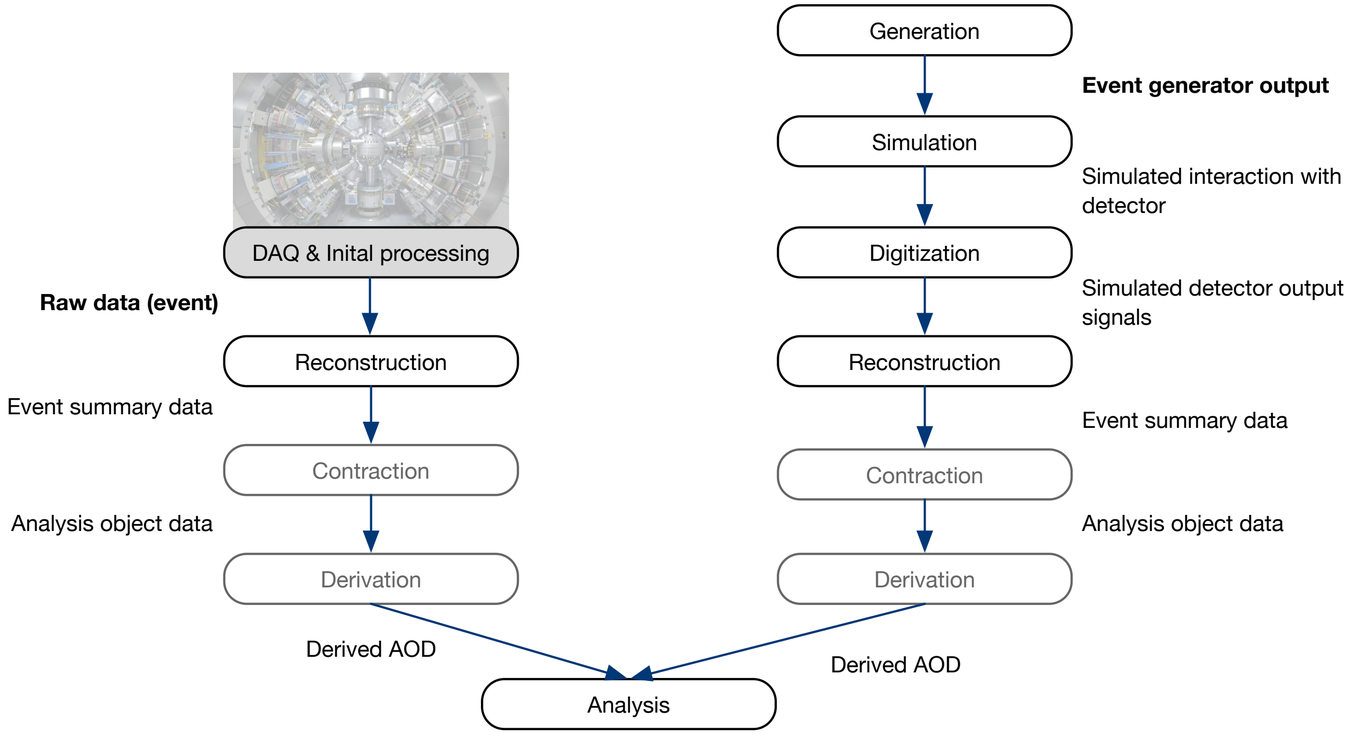
Data processing workflows

- A data processing workflow can be defined as a pipeline of discrete steps, where each step produces a distinct data product / "checkpoint".
- This modular, checkpoint-based architecture reduces the complexity of individual processing stages, enhances the reliability of the overall system, and improves the efficiency of computational resource usage.
- However, this approach also introduces significant challenges, particularly by increasing the demands for robust and efficient data management, especially within distributed computing environments.

Data processing workflows in HEP

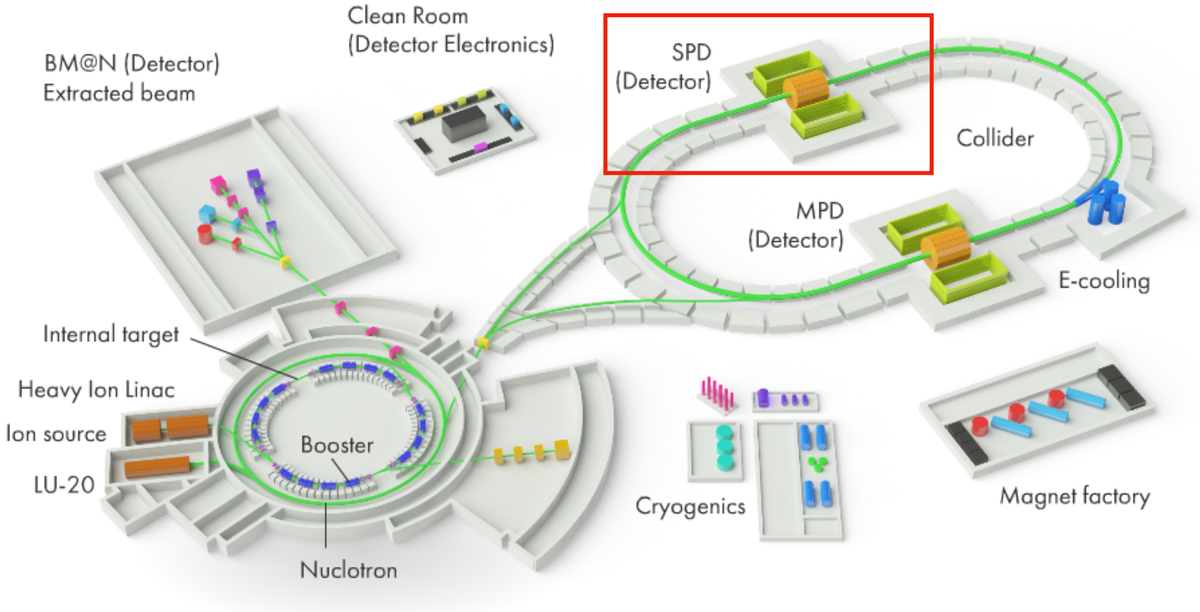
High-throughput computing (HTC) involves running many independent tasks that require a large amount of computing power.

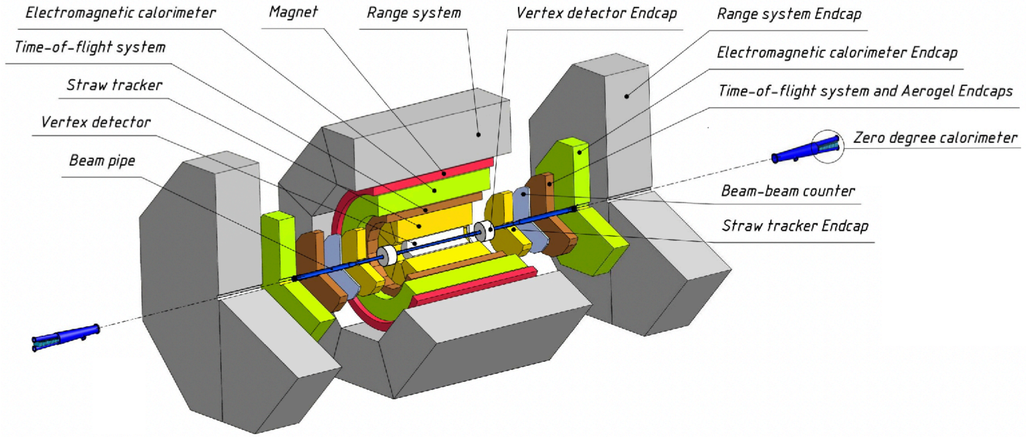
- Both Reconstruction and Simulation – are multistep workflows
  - Each step produces own data type, which correspond to different representation of events
  - So size of event will be different in different data type
- Why we need different types?
  - Some types of processing, like raw data, quite expensive or unique, producing of other types is resource consuming, another types good for long term storage but not optimal for final analysis because of redundancy



SPD Spin Physics Detector

Study of the nucleon spin structure and spin-related phenomena in polarized *p-p*, *d-d* and *p-d* collisions





- SPD - a universal facility for comprehensive study of gluon content in proton and deuteron

SPD as a data source

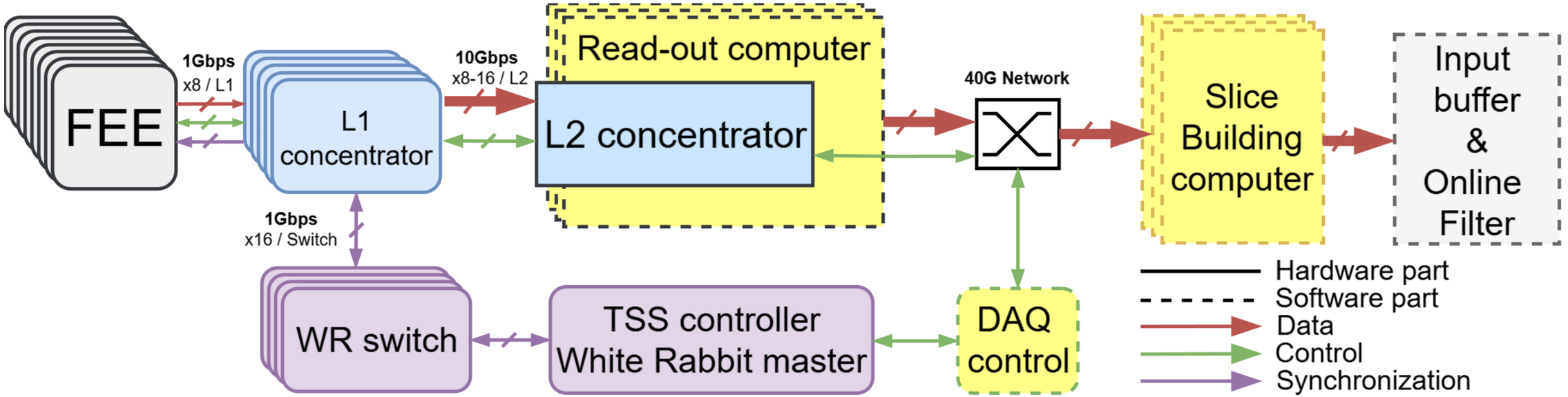
The SPD detector is a medium scale setup in size, but a large scale one in data rate!

- Bunch crossing every 76,3 ns = crossing rate 13 MHz
  - ~ 3 MHz event rate (at 10^{32} $cm^{-2}s^{-1}$ design luminosity)
- **20 GB/s** (or **200 PB/year** "raw" data, **~3*10^{13}** events/year)
  - “Only" **~1,5*10^{12}** events/year are interesting for detailed study (~10PB/year) of data
  - Selection of physics signal requires momentum and vertex reconstruction → no **simple trigger** is possible
- Comparable amount of simulated data

SPD DAQ Structure

- L1 concentrator
- L2 concentrator
- TSS – synchronization system
- Computer Read-out and Slice Building system.

SPD Stage	FEE Outputs	L1 (8 ports)	L2 (16 ports)	Read-out PC	Slice Builders
First	~1292	>165	>12	>6-12	>5
Second	~5800	>730	>50	>25-50	>40

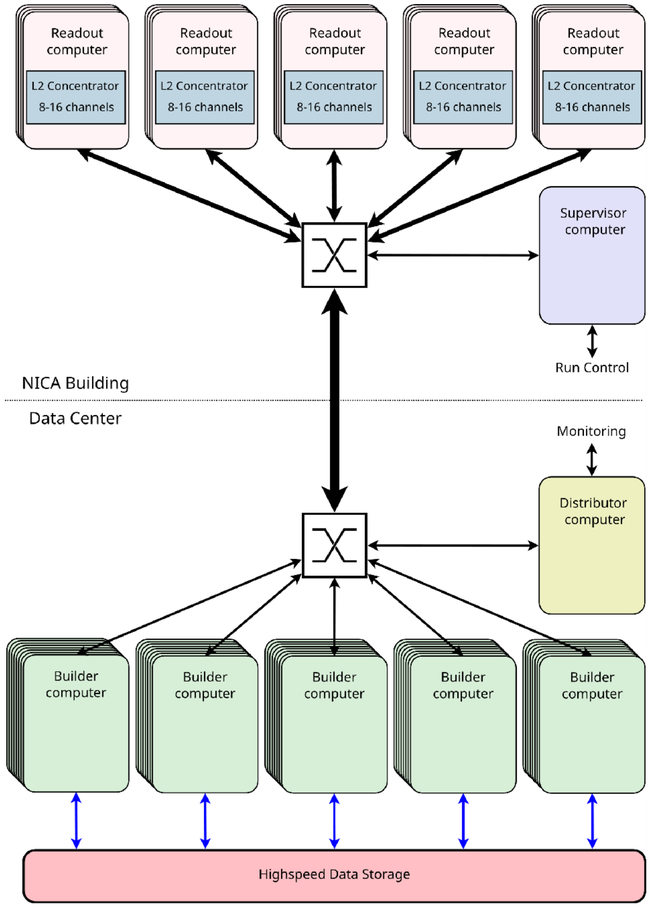


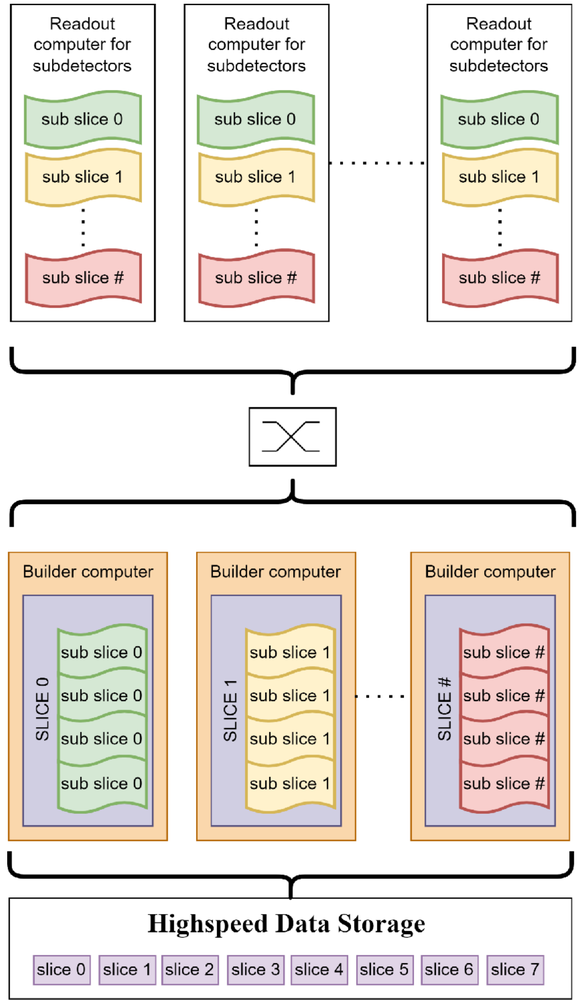
SPD initial data organisation

The purpose of slice building systems is to collect data (sub-slices) from all readout computers, build a complete slice and write it to the fast intermediate storage.

Components:

- Read-out – getting a sub-slice from each detector system
- Builder – combining sub-slices and forming a single data structure
- Supervisor – arbitration of the slice construction system
- Distributor – provides a monitoring data

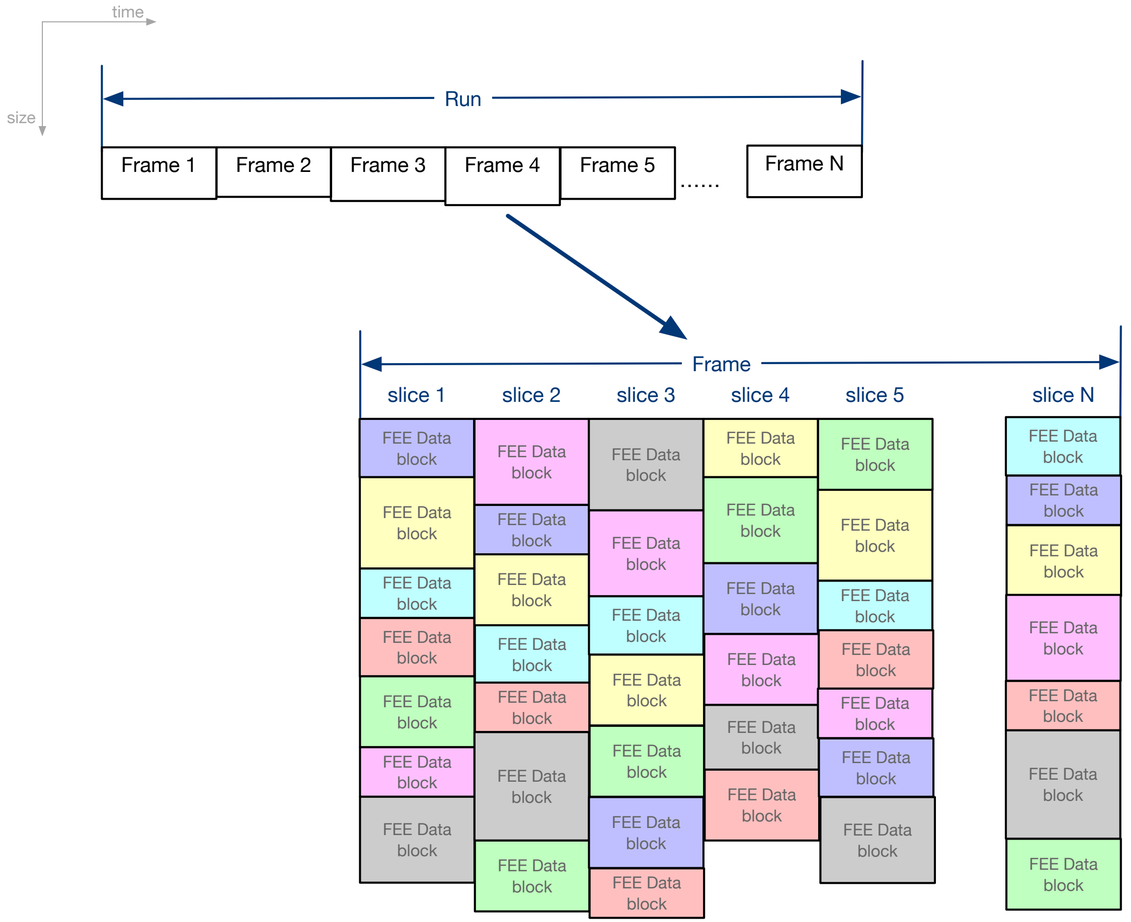




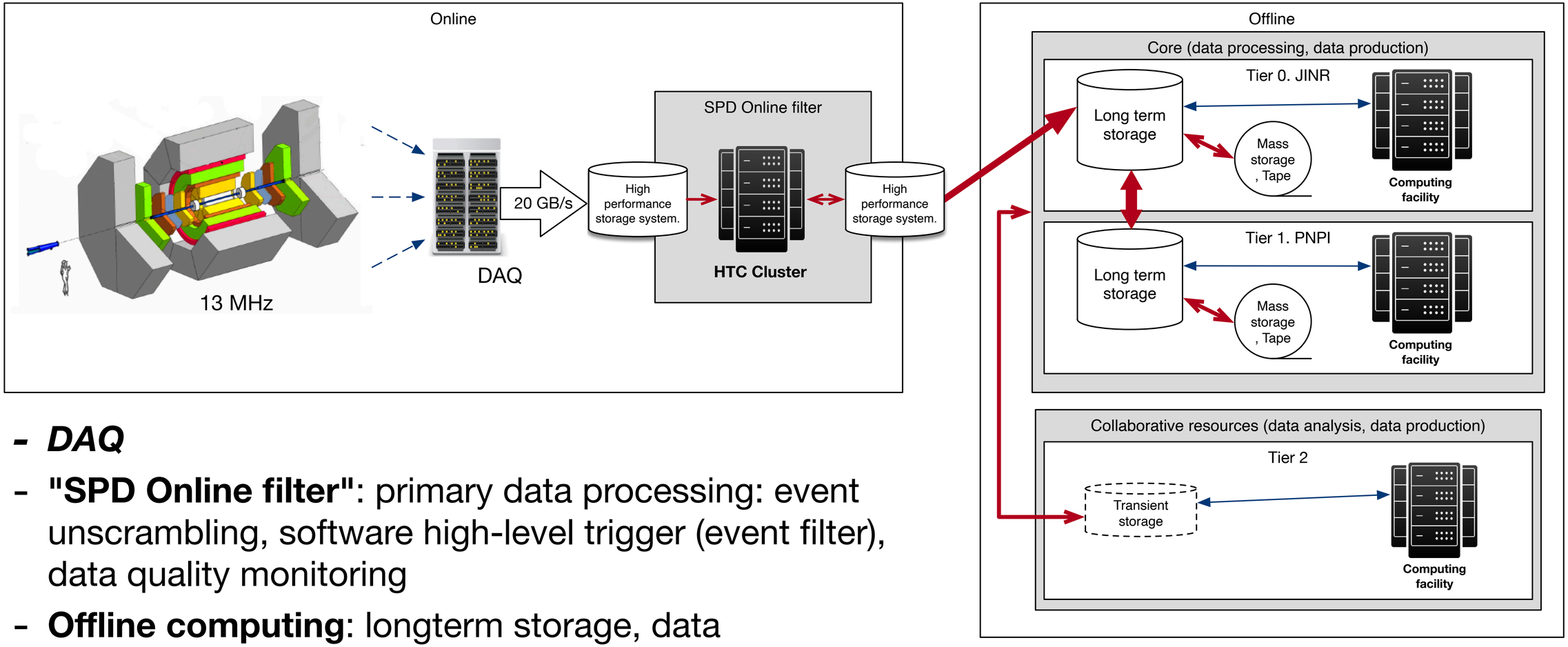
Free run DAQ and (raw) data format

No trigger = No "classical" events at the start

- Free run DAQ, means that the output of the system will not be a dataset of raw events, but a set of frontend electronics (FEE) data blocks from detectors organized in time slices
- Primary data unit: time slice (1 μs — 8.3 ms) Time slices combined in time frames (up to 549 s, 16 GB max, < 160 MB to fullfil 20 GB/s limit)
- Each FEE data block have sub-detector specific format and may contain signals from a few collisions (events)



Data flow



- ***DAQ***
- **"SPD Online filter"**: primary data processing: event unscrambling, software high-level trigger (event filter), data quality monitoring
- **Offline computing**: longterm storage, data production, data processing and analysis

SPD Online filter

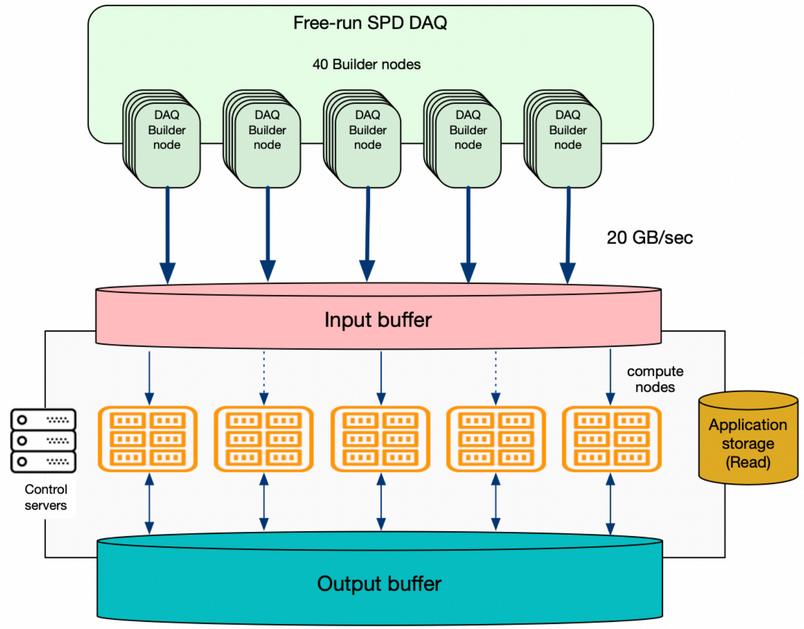
SPD Online Filter

Online filter is the first stage in data processing chain for SPD Experiment (right after DAQ)

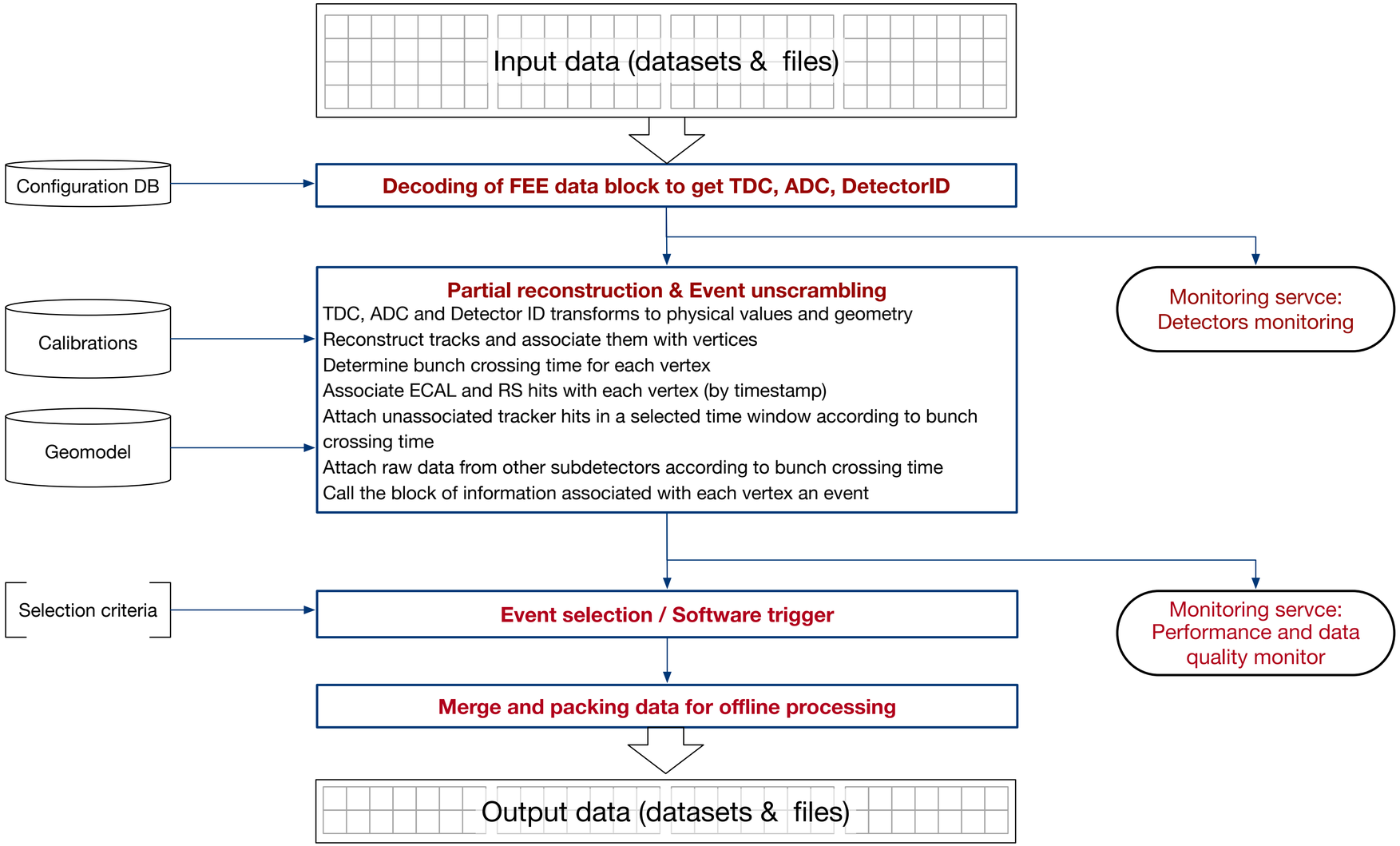
Main goals:

- Events unscrambling through partial reconstruction
- Software trigger, which essentially is event filter

- SPD Online Filter is a high performance computing system for high throughput processing
  - **Hardware component:** compute cluster with two storage systems and set of compute nodes
  - **Applied software**: performs informational processing of data. Had to use same framework as ‘offline’ applied software
  - **Middleware component:** software complex for management of multistep data processing and efficient loading (usage) of computing facility.



Data processing workflow on SPD online filter



Middleware functionality

Data management;

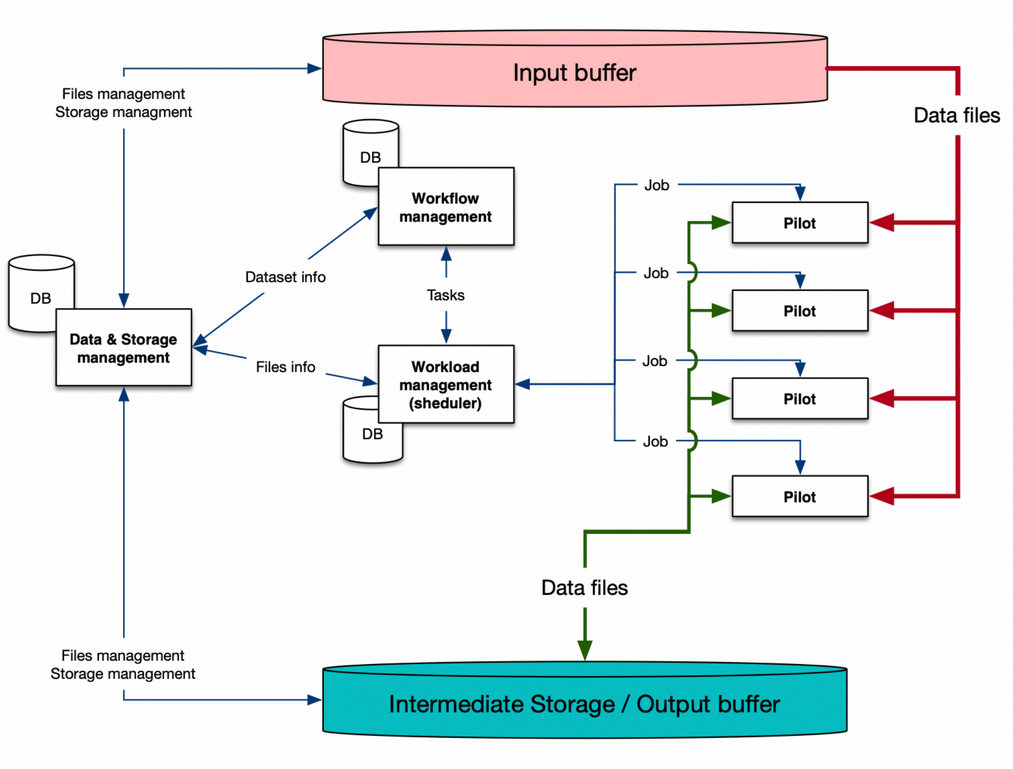
- *Support of data life-cycle and storage usage;*

Workflow management;

- *Definition of processing chains;*
- *Realisation of processing chains as set of computations tasks;*
- *Management of tasks execution;*

Workload management:

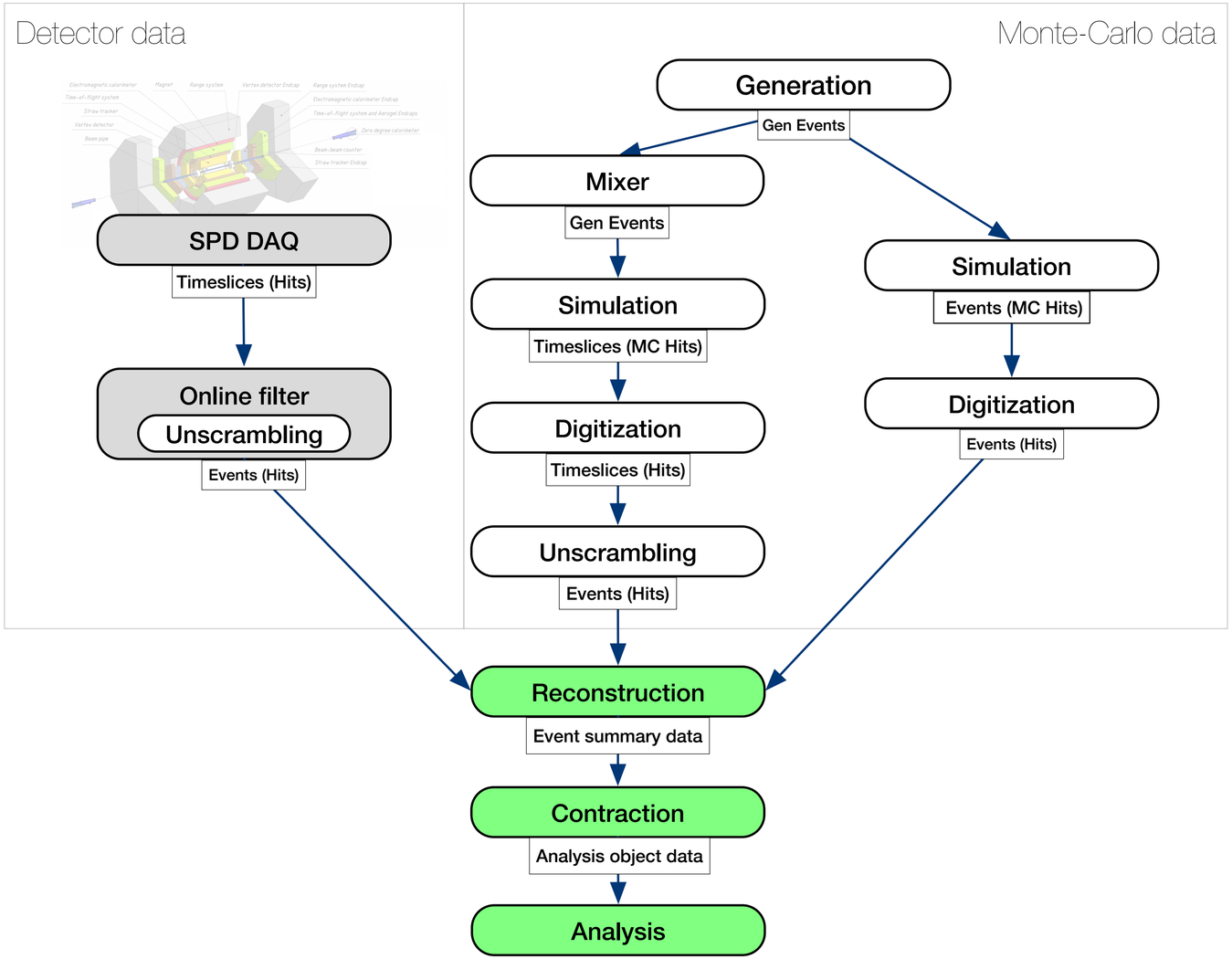
- *Generation of required number of processing jobs for performing of task;*
- *Control of jobs executions through pilots, which works on compute nodes;*



SPD Offline data processing

Timeslices simulation

- Free run DAQ increased complexity of data production (Monte-Carlo)
- Raw data in timeslices should be simulated along with events simulation



What kind of simulation do we need?
(1 of 2)

- Naive approach: the aim of the Monte-Carlo simulation is to imitate reality as precise as possible. So, we need to simulate only timeslices and use it everywhere
- **But:**
  - One timeslice is expected to have in average 1-2 potentially interesting events selected by an online filter
  - One timeslice is expected to have in average ≤0.001 signal events

What kind of simulation do we need?
(2 of 2)

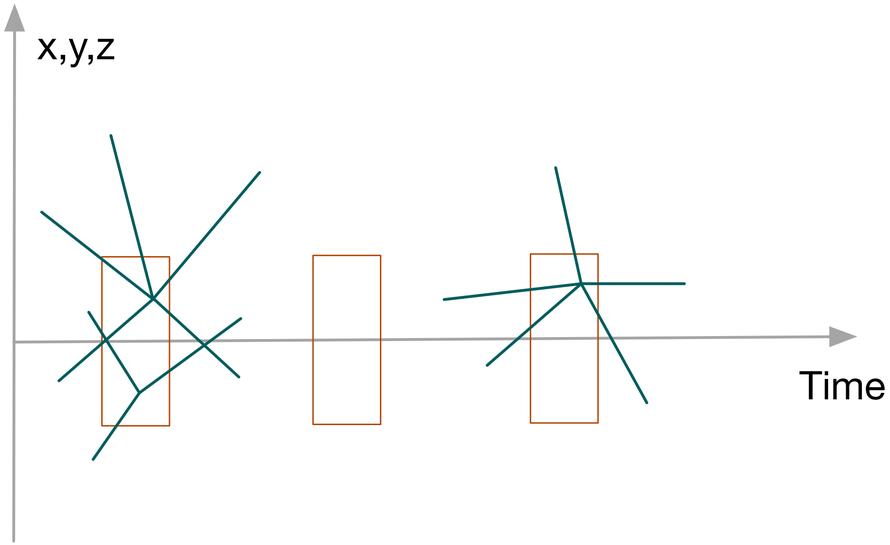
- **Simulation of events**
  - for the detector optimization and physics analysis
  - main simulation stream!
- **Simulation of timeslices**
  - Online filter optimization and tuning of the online reco algorithms (+ML training)
  - A rough estimate of selection efficiency at the online filter

Two simulations with the same software (and the same manpower!)

Timeslice Building

Given: timeslice duration, energy(=Integral cross-section), beam parameters

- Determine the number of bunch crossings (BC)
- Simulate a random time offset for the first BC
- Determine the number of collisions in the first BC (Poisson)
- Read the required number of generated events, set their time stamp and simulate their vertex position (x,y,z)
- Continue while all bunch crossings are simulated



Event simulation

Data model

Generator

```
Input: none
Output: HepMC::GenEvent
```

Detector simulation

```
Input: HepMC::GenEvent
HepMC::GenEvent → McParticle
Output: [McParticle] [McSteps]-optional [McHits] [EventHeader]
```

Digitization

```
Input: [McParticle] [McHits][EventHeader]
Output: [McParticle] [Digi][EventHeader]
```

Timeslice simulation

Data model

Generator

```
Input: none
Output: HepMC::GenEvent
```

Timeslice building

```
Input: [HepMC::GenEvent]
HepMC::GenEvent→ McParticle
Output: [McParticle]
```

Detector simulation

```
Input: [McParticle]
Output: [McParticle] [McSteps]-optional [McHits] [TimesliceHeader]
```

Digitization

```
Input: [McParticle] [McHits] [TimesliceHeader]
Output: [McParticle] [Digi] [TimesliceHeader]
```

Online Reco

```
Input: [McParticle] [Digi] [TimesliceHeader]
Output: [Event]
Each event: [McParticle] [RecHit] [FastRecObjects: Tracks, Clusters...] [EventHeader]
```

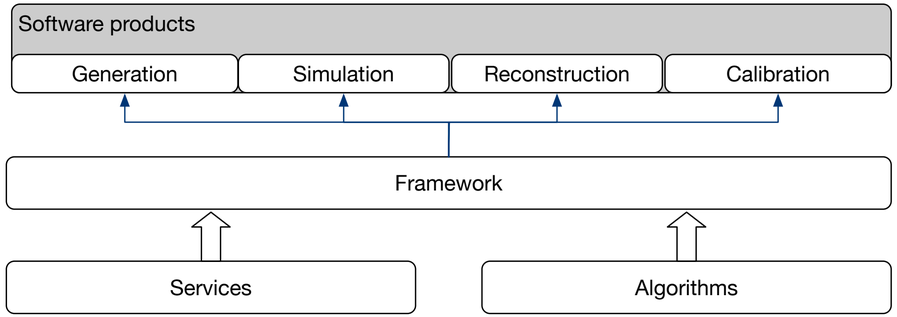
SPD Software framework - Sampo

Information processing of physics data of SPD Experiment

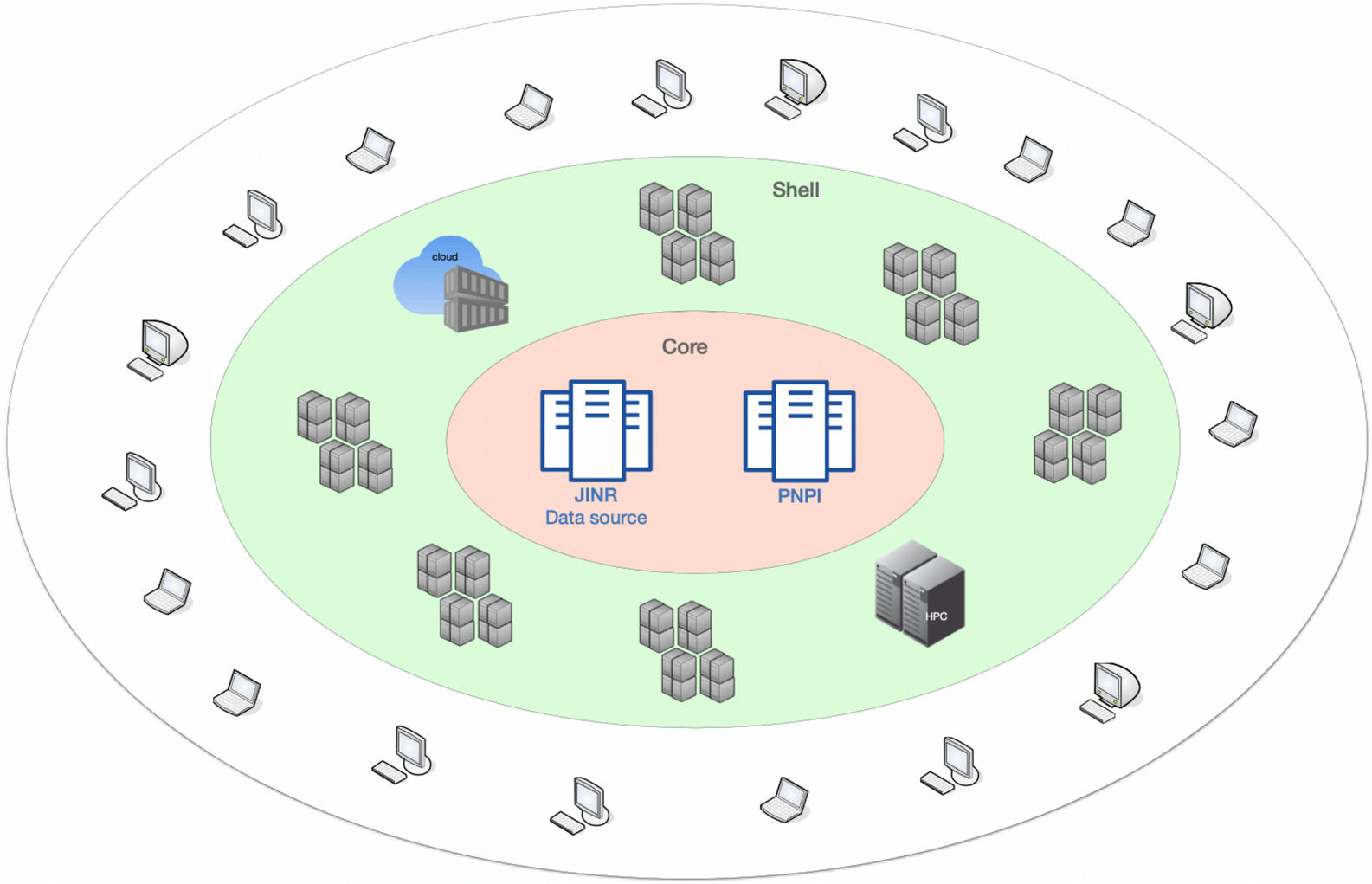
- Based on Gaudi:
  - Intra- and Inter-Event parallelism
  - Configuring via python
  - Easy to develop new components (established architecture)
- Current functionality
  - Pythia8 for events generation (multithreaded)
  - GeoModel for detector description
  - HepMC3 as main format, HepMC2-compatible
  - Geant4 for simulation (multithreaded)
  - JobConfigurator for job configuring

Current simulation and performance studies are done by another framework SpdRoot, based on FairRoot software

- Does not fit well for massive data processing



SPD Offline computing system



- **Core sites (JINR, PNPI)** – data long term storage, main data processing and producing
- **Shell sites** – data analysis, data producing

Thank you!


С САМОГО НАЧАЛА
У МЕНЯ БЫЛА КАКАЯ-ТО
ТАКТИКА
И Я ЕЁ ПРИДЕРЖИВАЛСЯ

A few more definitions

- **SPD Software** - a set of activities related with development, support and evolution of applied software for **information processing of physics data of SPD Experiment**.
- **SPD Computing** - a set of activities devoted to setup and operation of distributed computing environment for **data processing of SPD Experiment**.
  - *Usually we call it - distributed data processing*
- **Infrastructure** - a set of computing and storage resources provided by collaboration members for shared usage in distributed data processing