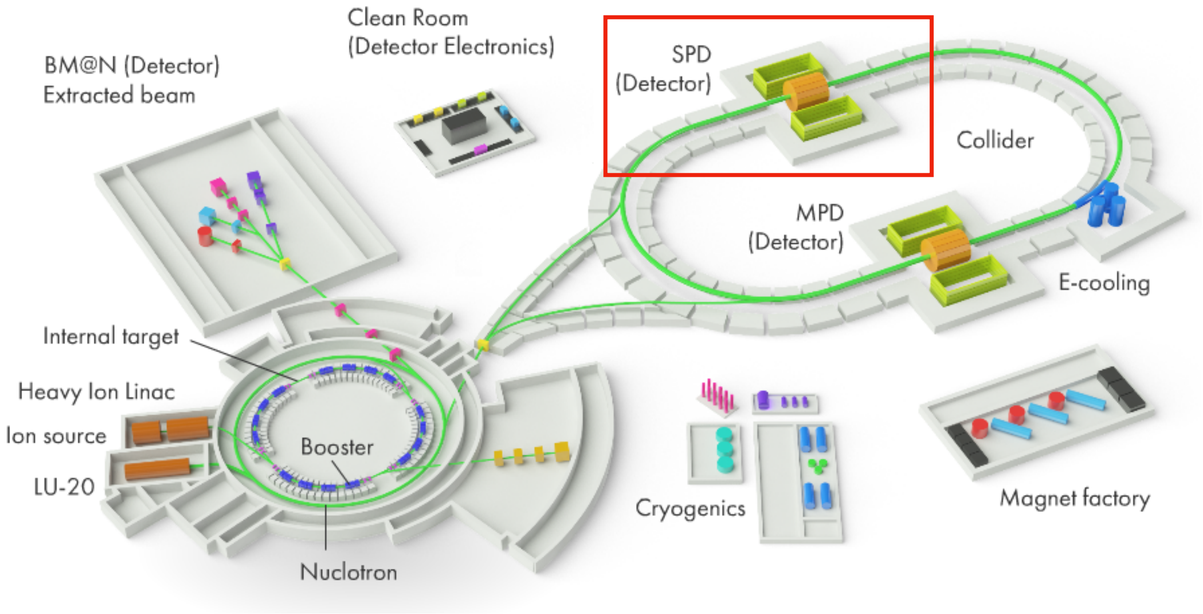
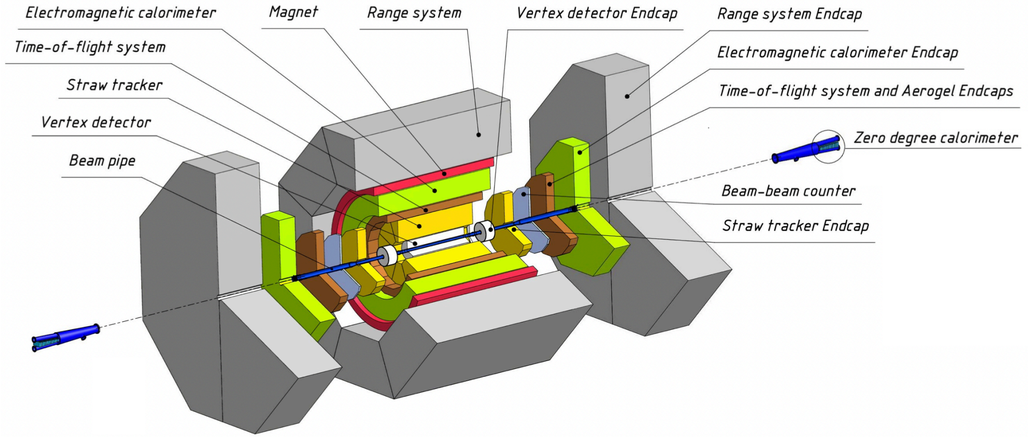
Computing challenges for SPD experiment at NICA.

Danila Oleynik
danila.oleynik@jinr.ru

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 data source

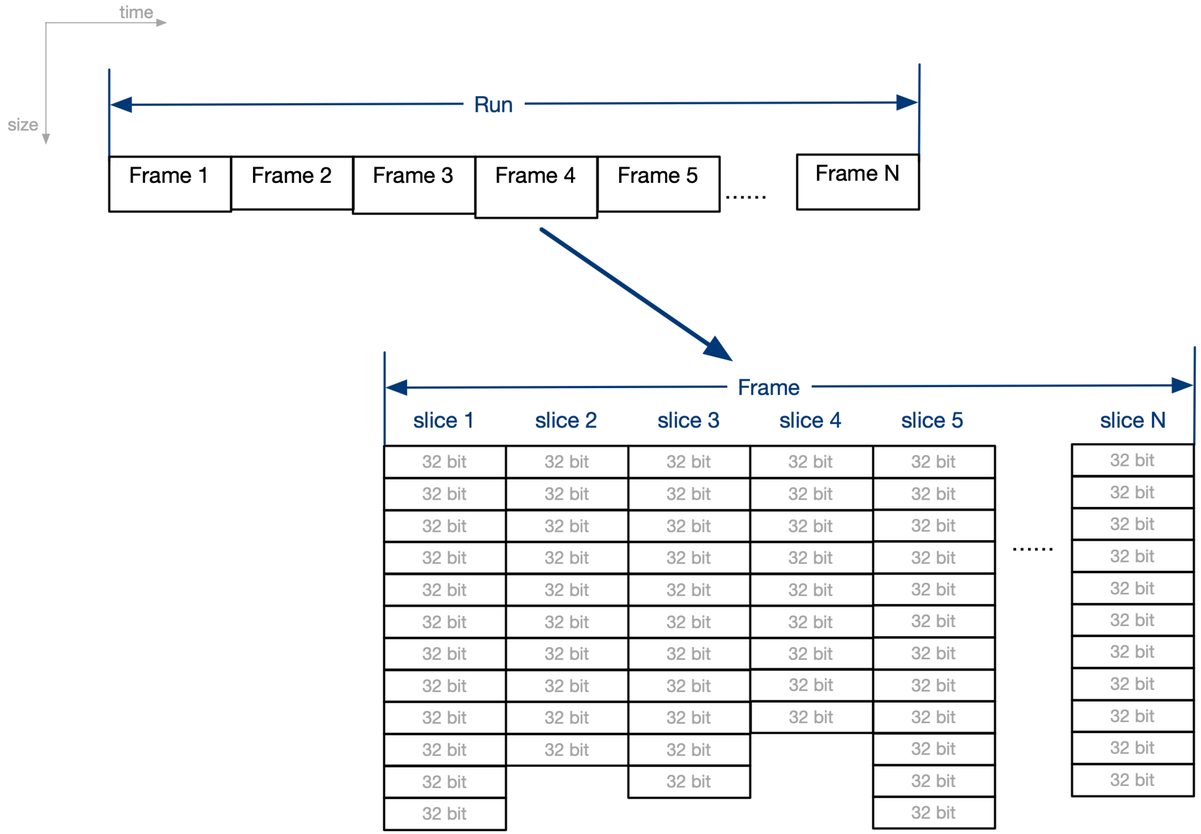
- Bunch crossing every 76,3 ns = crossing rate 13 MHz
- ~ 3 MHz event rate (at 10^{32} $cm^{-2}s^{-1}$ design luminosity) = **pileups**
- **20 GB/s** (or **200 PB/year** "raw" data, **~3*10^{13}** events/year)
  - Selection of physics signal requires momentum and vertex reconstruction → no **simple trigger**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

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

Challenge 1: initial data processing

Free run DAQ and data format

- Free run DAQ, means that the output of the system will not be a dataset of raw events, but a set of signals 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)
- Intermediate units — time chunks of 0.1-0.2 s (2-4 GB or ~10^5- 10^6 events) are being discussed now
- Every time slices will contain signals from a few to many collisions (events)
- Event building have to unscramble events from a series of time slices



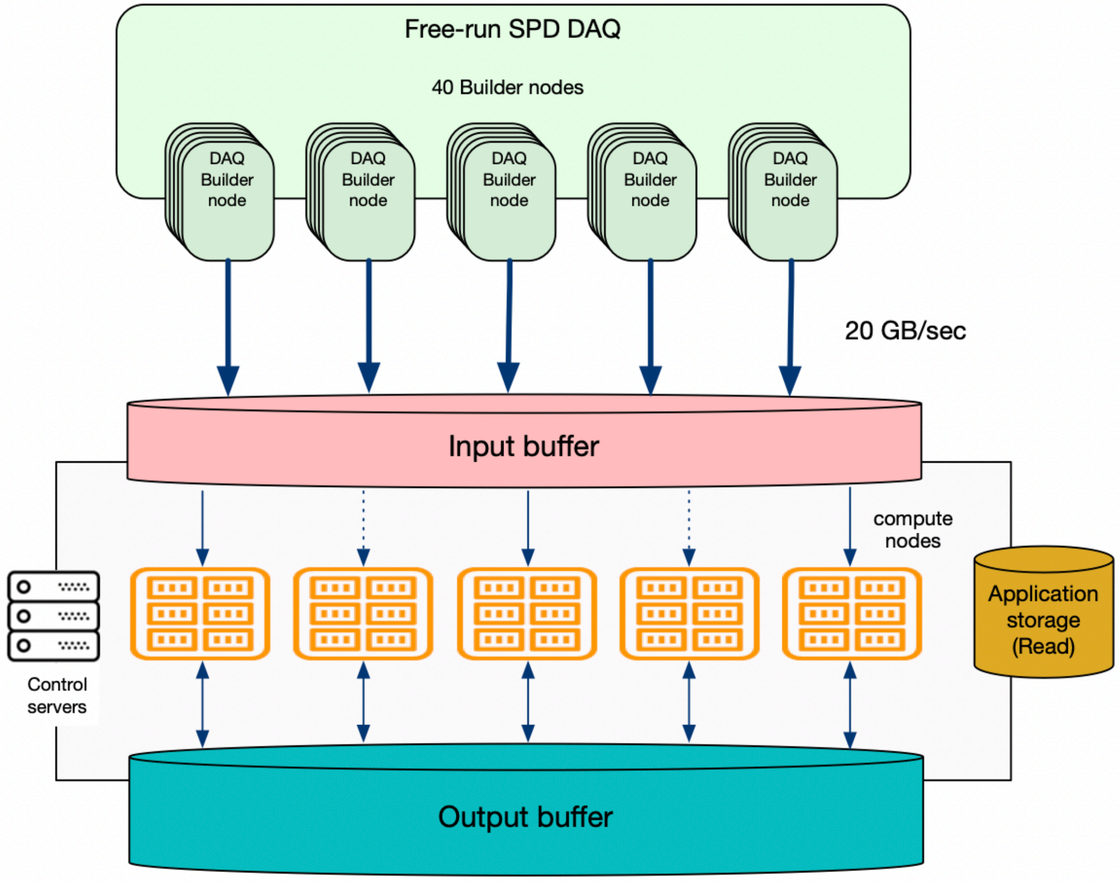
Event unscrambling

For each time slice

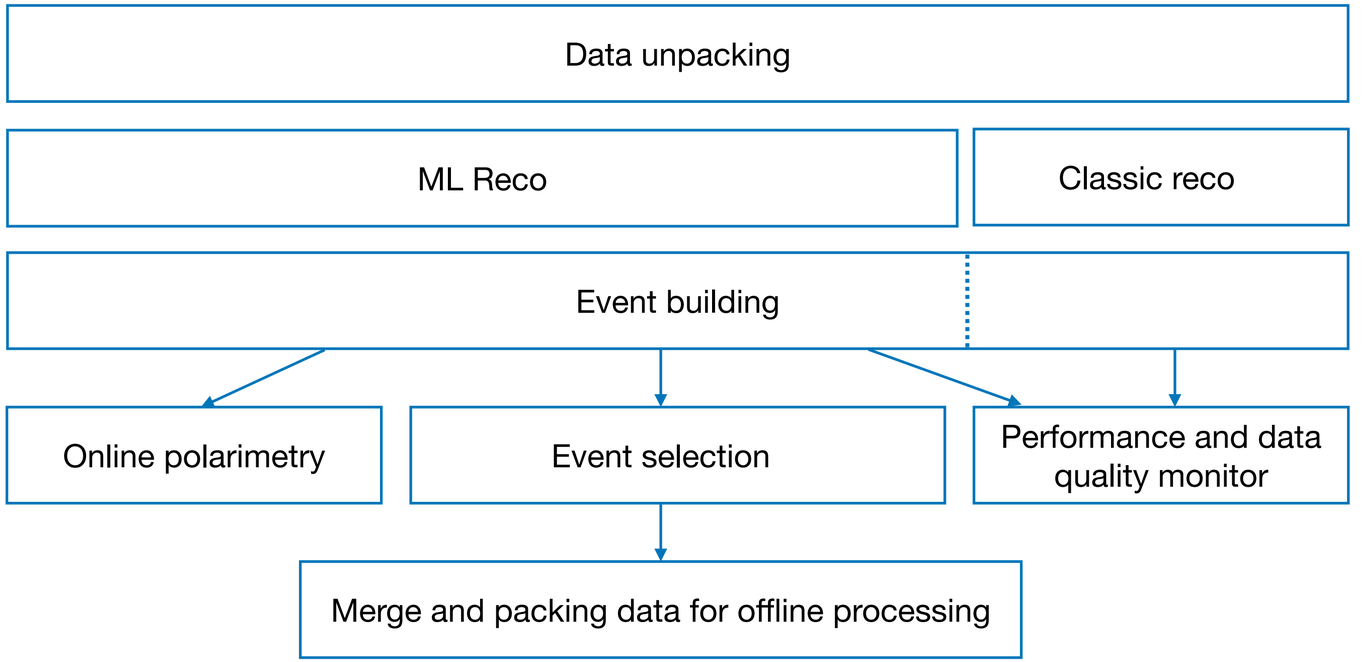
- Reconstruct tracks and associate them with vertices
- Determine bunch crossing time for each vertex
- Associate ECAL and RS hits with each vertex (by timestamp)
- Attach unassociated tracker hits in a selected time window according to bunch crossing time
- Attach raw data from other subdetectors according to bunch crossing time
- Call the block of information associated with each vertex an event
- Store reconstructed events

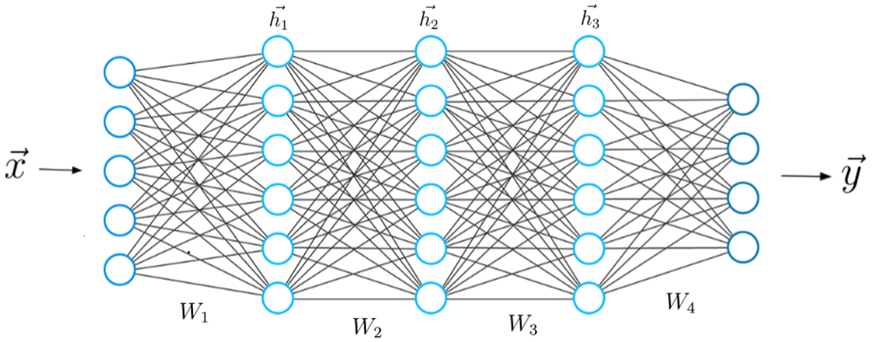
SPD Online Filter

- SPD Online Filter is a high performance computing system for high throughput processing
  - High speed (parallel) storage system for input data written by DAQ.
  - Compute cluster with two types of units: multi-CPU and hybrid multi CPU + Neural network accelerators (GPU, FPGA etc.) because we are going to use AI…
  - A set of dedicated servers for middleware which will manage processing workflow, monitoring and other service needs.
  - Buffer for intermediate output and for data prepared for transfer to long-term storage and future processing.



Payload





Machine learning is a promising technology

Middleware required functionality

Data management;

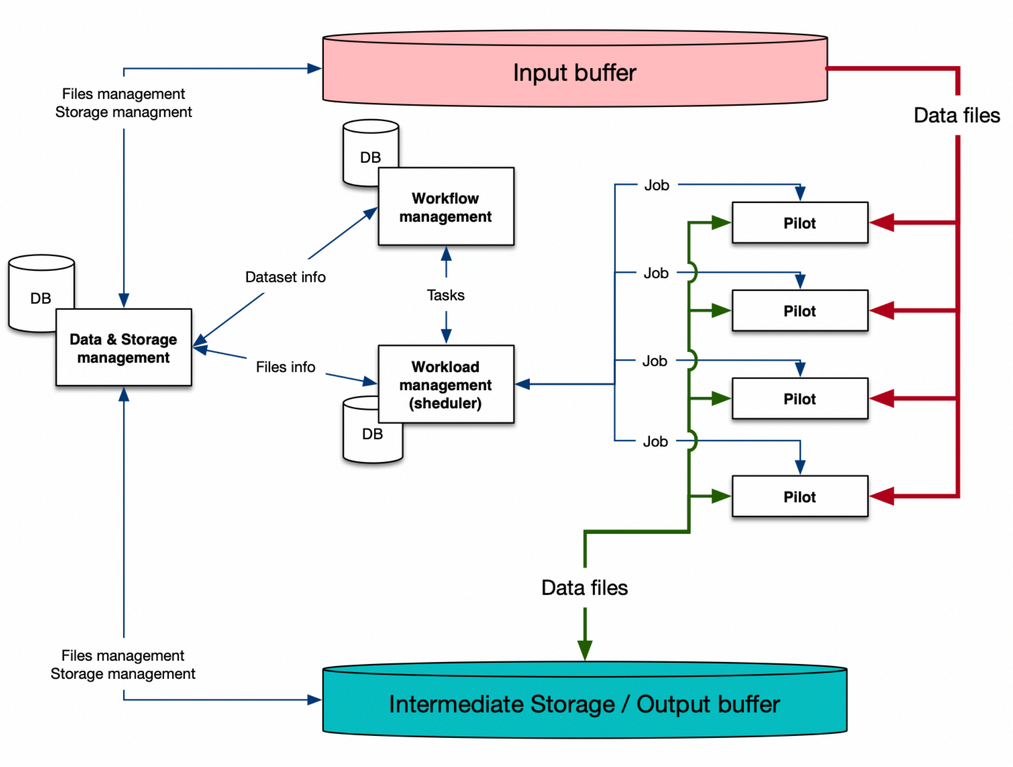
- *Support of data lifetime (registering, global transfer, cleanup);*

Processing management;

- *Generate jobs for each type of processing:*
  - *Data unpacking*
  - *Partial reconstruction;*
  - *Verifying of processing results (AI vs traditional processing);*
  - *Select (Filter) events;*
  - *Pack (merge) output data for transferring to “offline”;*

Workload management:

- *Dispatch jobs to pilots;*
- *Control of jobs executions;*
- *Control of pilots (identifying of "dead" pilots)*



Workflow management

Curent status

- Workflow management system is the high level system which interacts with data management system and workload management system for realising multistep processing of blocks of data (datasets)
- The functional decomposition of the subsystem was carried out and the initial set of microservices was proposed
  - “Chain definer” - user oriented application which allow define sequences of processing steps
  - “Processing starter” - microservice responsible for triggering of processing chains
  - “Chain executor” - microservice responsible for control of execution of processing chain.
- In progress, coding of interfaces with external systems, tuning of state model.

Data management

Current status

- A lot of work already done for data management system starting from decomposition of functionality to microservices, definition of set of tools for storage management, realisation of DB design for data catalogue (datasets and files) up to definition and realisation of the set of internal and external interfaces
- Microservices:
  - dsm-register – responsible for registration of input data from DAQ in the catalogue
  - dsm-manager – realise interfaces to the catalogue for subsystems
  - dsm-inspector – realise auxiliary tools for storage management (consistency check, cleanup, dark data identification)

Workload management

Current status

- Realize a task execution process by shredding a required number of jobs to provide controlled loading to compute facility, tacking into account priority of tasks and associated jobs. A task is one step in a processing chain of a block of data. Job is a processing of a single piece of data (file or few files).
- Microservices: task manager, task executor, job manager, job executor
- Workload management system intensively interacts with Pilots, Data Management system, accept tasks from Workload management system and reports progress of execution back
- Base architecture in place, coding of internal and external interfaces.

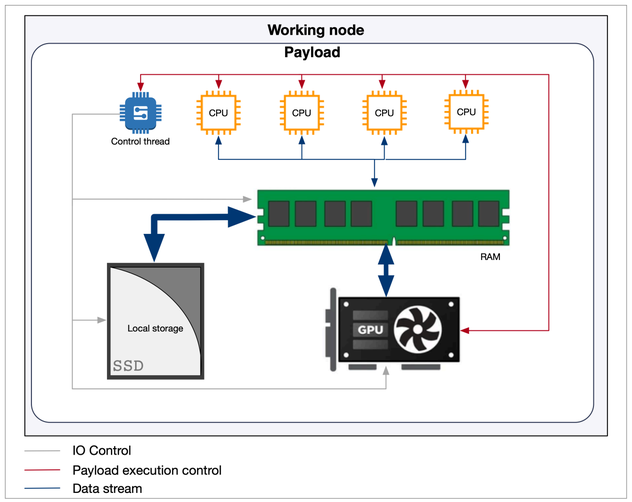
Pilot

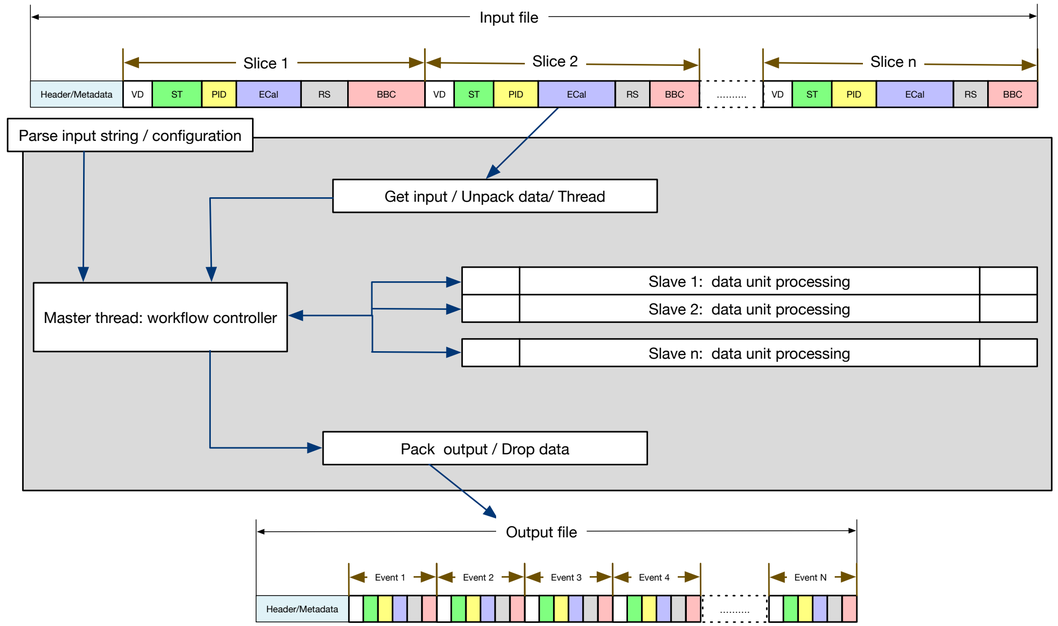
Current status

- Pilot is the application to manage the execution of a job, produsedby WMS, on a compute node. It responsible for setting up of environment, stage-in/out of data from storage to compute node, execution of payload and monitoring of different conditions during execution. Intensively interacts with workload management system
- Base architecture and initial functionality of pilot application is defined. It is a multithread application with interactions between threads through queues
- In progress intensive development of interfaces and initial definition of internal entities

Challenge 2: Multithread processing

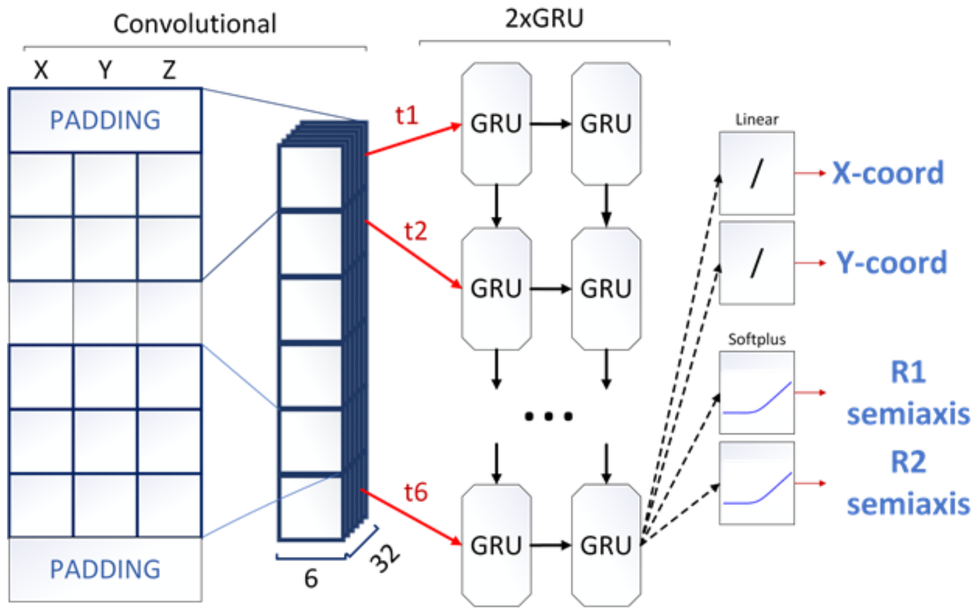
- Multicore computers already reality ;-)
  - Efficient usage requires multithreading processing
  - A lot of algorithms in HEP software stack does not support multithread execution (yet)
- We tries to explore multithread processing on data layer (each thread process own piece of data)





Challenge 3: Machine learning

Example: TrackNETv3 for track recognition



- Network predicts an area at the next detector layer where to search for the track continuation
- If continuation is found the hit is added to the track candidate and the procedure repeats again
- Essentially reproduces the idea of the Kalman filter: track parameters are predicted by synaptic weights determined by network training
- *Generalization? Stability?*

	Time slices of 40 events
Track efficiency (recall) (%)	96,54
Track purity (precision) (%)	94.75
Time slices / sec	63.74 (*40 = 2549.6)

Machine learning: from R&D to production

- New production chain: producing of trained neural networks
  - In general, each produced NN should be 'linked' with training sample from one side and with results which will be getting with a particular version of network
    - New data types, proper storing and catalogue will be required
- Training is quite compute-intensive operation, so usage of HPC resources should be taken into account
  - with high priority during data taking

Challenge 4: Information systems & databases

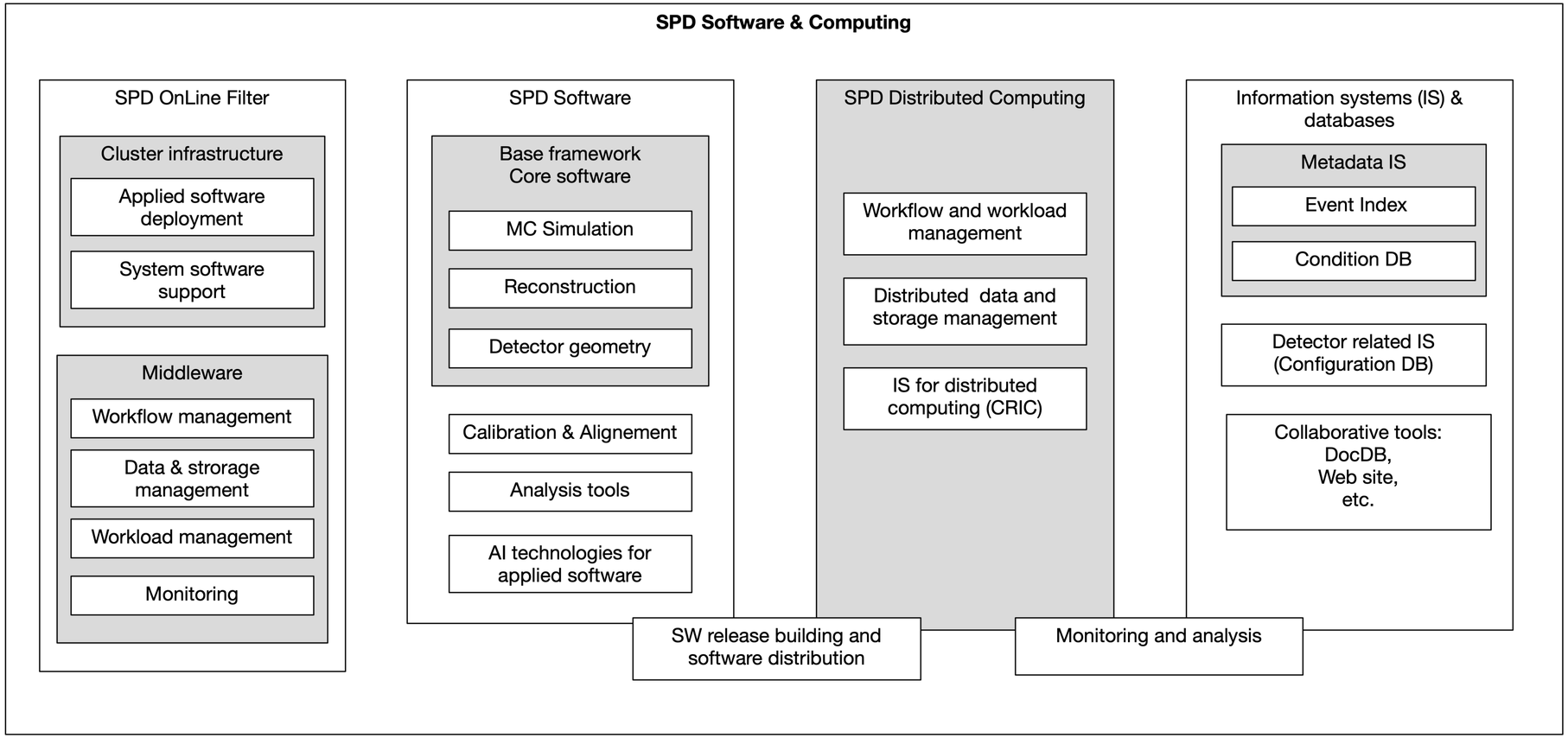
- Event index - is the set of special information systems which allows to store and navigate across all produced events
  - In simple words Event index allows identify dataset or even file where particular event is stored.
  - Quite important system as only you start to use hundreds of thousands files
- Condition database - stores data which is not related with event production itself, but status of environment during data tacking
- Configuration database - stores detector hardware setup and other hardware related information
- Detector hardware database (detector elements, cabling etc)
- Run database
- Offline DB: Geometry versions, Calib&Align, Magnetic field

Scientific digital services

- Collaborative tools:
  - E-log - digital shifter logbook
  - Project management and issue tracking system
  - Digital libraries for managing of documentation and publication
  - Group calendars, meetings support etc…

How challenges addressed?

SPD Software & Computing



It's painful, but computing involved quite deeply in the lifecycle of scientific research.

JINR MLIT role in SPD Software & Computing

What is “Tier 0” in reality?

- Obviously, JINR LIT should provide required Tier 0 data processing centre functionality for NICA experiments and it’s not only compute and storage resources:
  - development and supporting of common collaborative digital services;
  - authentication and authorization infrastructure, including support of auth for NICA distributed computing;
  - will be very nice to have central DBMS support as well as central hosting service

JINR LIT role in SPD Software & Computing

Not only infrastructure!

- Methodological support and R&D in the field of machine learning solutions
- Development, support and operation of middleware as for offline processing so for the SPD OnLine filter
- Participation in prototyping of SPD DAQ and the SPD OnLine filter

Summary

- SPD experimental apparatus requires as ‘traditional' so some new methodology of computing and software environment:
  - SPD Online filter for high-throughput initial data processing with efficient usage of heterogeneous multicore compute nodes
  - AI technologies, in particular ML for speed-up of initial data processing, brings a few new challenges
  - Full set of information systems and databases required for experiment
- SPD Software & Computing works close with JINR MLIT to address appeared challenges

Thank you!
Have a great evening!