SPD Information systems

Current status

SPD Information Systems

- The SPD Experiment have to produce large amounts of data, both collected from the detector and simulated
- The processing of the experimental data requires a wide variety of auxiliary information from many systems
- Huge numbers of the detector condition and management data should be stored in the databases
  - should be used in every stage of data taking, processing and analysis
  - are essential at nearly every stage of data handling
  - for use in number of versatile applications each with its own requirements

SPD Information Systems

- Databases can not be considered as the thing in itself, but as a part of complex information system that include
  - Data collection tools
  - Data transportation tools (messaging services, etc)
  - Application layer between the client and the database server, including caching proxies
  - Client software
  - APIs for access from the producton and analysis software
  - Supervisors and monitoring

SPD Information Systems

- Hardware Database
- Mapping Database
- Conditions and Calibration Databases
- Physics Metadata Databases
- Event Index
- Distributed Computing Information System
- Distributed Data Management
- Monitoring information system
- Logging and Bookkeeping
- Personal and publication databases

Hardware Database

- A catalog of hardware components that SPD detector consist of.
- It should contain the information about the detectors and the electronic parts, cables, racks, and crates, as well as the location history of all items
- It include equipment models, provider, parameters and other (semi)permanent characteristics
- This should help in maintenance of the detector systems and especially helpful in knowledge transfer between team members.
- A prototype system is being developed, including PostgreSQL as a back-end, that is bein accessed through the REST API from the web interface

Tables schema

HWID	type	S/N	Notes	state	parameters
0007-015b2e3488ac	04fd15c3	PG1342		OK	{JSON}
0007-015b2e3488ad	04fd15c3	PG1344		FUBAR	{JSON}
0007-01fe47adf301	0368eba1	164756		FU	{JSON}
0007-01fe47adf301	0368eba1	164756		OK	{JSON}

TypeID	Name	Description	parent	parameters
04fd15c3	PG2T390	PANGOMICRO Titan-2 PG2T390 FPGA Board	000013f0	{JSON}
0368eba1	EQR151110	EQR15 11-1010D-S SiPM	00001134	{JSON}
000013f0	Concentrator			
00001134	SiPM			


```
{
  ...
  "ov": "37",
  "dcr": "253",
  ...
}
```

```
{
  "rov": {
    "description":"rec. oper. V",
    "value":"38"
  },
  "ov": {
    "description":"oper. V",
    "type":"dec_1",
    "lo":"36",
    "hi":"40"
  }
  ...
}
```

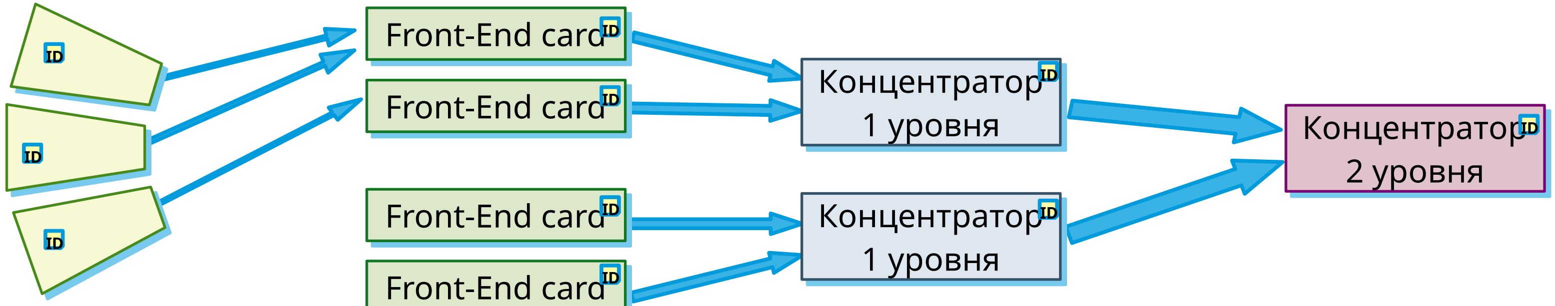
Hardware Database

- For each type of device, a set of parameters are defined that are common to all devices of this type,
  - Each parameter has type as well as optional ranges of allowed values
- There can be common values for all devices of the same type
- Each device has unique ID assigned to it, and specific values of the parameters can be specified for it, based on its type.

- A Input for the further development of the HWDB is required
- Most of the subsystems are in the very early stage of the development
- For some detectors, like Range System, design of the system and components are already defined to some extent
- The development of the HWDB can be based on the input from them

DAQ Mapping

- The number of data collection channels of the SPD installation will be several hundred thousand
- The signals from the detector will pass through several communication devices



- It is necessary to have a mapping of the data collection system that establishes the correspondence of the channel addresses at the DAQ outputs with the devices from which this signal came

Building of mapping

- Due to the large number of elements in the system, it is almost impossible to build mapping manually
- For the elements involved in the transmission of digital signals, an automatic mapping procedure should be implemented
  - The element must issue a HW ID over the data channel in response to a special signal
- For parts of the system that are not equipped with automatic source ID recognition, an interface must be provided that allows data entry by groups.

Performance requirements

- It is expected that the filling of the hardware database will take place gradually, and updates will be rare
- The construction of the connection diagram and its changes will also be performed rarely (no more than once a week)
- The requirements for the speed of recording information in the database are low
- Mapping information may be required when processing each file. It is possible that tens of thousands of processes will try to simultaneously access the system.
- It is necessary to ensure their processing, avoiding database overload due to too high frequency of requests

Conditions and Calibration Data

- Conditions data - non-event data representing the detector status
  - Detector hardware conditions:
    - Temperatures, currents, voltages, gas pressures, etc.
  - Detector read-out conditions:
    - Trigger and detector read-out configurations
  - Detector calibrations:
    - Energy calibration for calorimeters, time-over-threshold for pixels, etc.
  - Detector alignments:
    - Relative and global alignment of sub-detectors
  - Physics calibrations:
    - Energy scales and resolutions, reconstruction efficiencies
  - Luminosity and polarization measurements:
    - Roughly measured during run, precise values based on event reconstruction

Conditions data usage

- Subsystem calibration
  - conditions determination and testing, uploading to the production DB
- Online data processing
  - using the conditions data that were declared as valid in the past
  - changes are deployed less frequently.
    - major conditions updates only after the experts have fully checked the data.
- Primary data processing
  - condition data can be updated in the progress based on the “express” sample of reconstructed data
- Reprocessing Campaign
- User analysis
  - Various pieces of conditions data may be needed for some analyses

Conditions Data organisation

- Various pieces of information are heterogeneous both in data type and in time granularity
- The data should be organized by "Intervals of Validity" (IOV), which is the span in time over which that data is valid
- Can be recomputed later if the understanding of the detector behavior improves or the quality of the input data increases.
  - except for the detector and trigger configurations
- Careful versioning of groups of conditions data for production use cases is a critical item to guarantee reproducibility.
- Conditions data are typically written once and read frequently
  - read-rates up to several kHz must be supported for distributed computing use

Condition data with REST

- We consider adopting CREST (Condition data with REST) project
  - developed for ATLAS experiment condition data with strong contribution of JINR developers
- CREST supports validity intervals and versions for all types of condition data
  - CREST architecture is designed as a client-server model, with a relational database as a backend
  - Data access was implemented using a pure REST API with JSON support.
  - The C++ Client Access library provides an interface for HTTP requests
  - Caching query results using a SQUID proxy can significantly reduce the load on the database

Distributed Computing Management

- **Distributed Data Management (Rucio):**
  - Dataset contents catalogue: list of files, total size, ownership, provenance, lifetime, status etc.
  - File catalogue: size, checksum, number of events
  - Dataset location catalogue: list of replicas for each dataset
  - Data transfer tools: queue of transfering datasets, status etc.
  - Deletion tools: list of datasets (or replicas) to be deleted, status etc.
  - Storage resource lists, status etc.
- **Production and Distributed Analysis (PanDA)**
  - Lists of requested tasks and their input and output datasets, software versions etc.
  - Lists of jobs with status, running locations, etc.
  - Lists of processing resources with their status etc.

Physics Metadata

- Contain information about
  - Datasets and data samples,
  - Provenance chains of processed data with links to production task configurations,
  - Cross-sections and configurations used for simulations,
  - Online filter and luminosity information for real and simulated data.
- Should collect a great part of its information from other IS’s and provide links to data there
- As MC data generation started, development of this IS is actual
- Conventions for the runs and dataset naming have to be defined, as well as for software and MC configuration versioning

Event Index

- Event Index is a system designed to be a complete catalog of SPD events, real and simulated data
- SPD Event Index is being developed as a comprehensive information system that should provide:
  - obtaining information about experimental events and simulated data by indexing data files containing information about these events;
  - transfer of this information and write to databases;
  - access to information for data processing and analysis programs via API and applications;
  - access to information to users through interactive and asynchronous interfaces.

Event Index Event Record

- An entry for an event in EventIndex must contain the following fields:
  - Event IDs: Run number (`run_number`) and event number in Run (`event_number`)
    - For the simulated events a field similar to the Run number should be implemented
  - Information about online filter solutions, in the form of a bit mask (`olf_result`)
  - Unique identifier of the RAW data file containing this event (`fuid_raw`). Using the UUID of a file, you can access it through a distributed storage system.
  - ID of the dataset this file is included in (dsid_raw)
- As the data is processed, new instances of the recovered events will be created in a format optimized for physical analysis (AOD).
  - Pointers to different versions of such files will be added to the event record.
  - Also, important event parameters can be added to the record, which can be used for classification and selection.

Event Index back-end

- The estimated data flow at the output of the online filter will be
  - from tens thousand events per second in the early stage
  - up to 150 thousand at maximum machine performance
  - from hundreds of billion (10^{11}) to few trillion (10^{12}) events per year.
- A system capable of coping with a flow of hundred of thousands of events per second is needed
- PostgreSQL DBMS is used for storage and processing of data:
  - ability to process large amounts of data, multithreading
  - high performance in data ingestion by support of bulk loading
  - open source, widely used, employed in other IS of NICA experiments
- Various optimization methods were investigated to speed up the process of writing data to tables

Event Index API

- A convenient and efficient program interface was developed and implemented that performs data exchange using the RESTful API.
- The front-end part of the client interface was developed using the Angular framework
- For the server side was chosen a fastAPI: a light weighted asynchronous RESTful framework for Python
- For asynchronous task processing RabbitMQ and Celery were used to improve system performance.
  - RabbitMQ is a message broker that allows you to send, receive and route messages between application components asynchronously
  - Celery is a system for performing operations in the background.

Further development

- Further development of the Event Index project:
  - Development of user authorization and authentication, group access policies
  - API development (REST, Python, C++, ...)
  - Optimization of user request processing, with synchronous or asynchronous output of results, depending on the volume of requested data
  - Development of mechanisms for transmitting "EventIndex" data obtained by indexing files located on remote nodes of a distributed computing network
  - Supervisor - software for managing, collecting and importing data
  - Development of an EventIndex component monitoring system, with graphical representation of data based on popular platforms (Grafana, etc.)
- The implementation of these tasks will be carried out in parallel with the development of other Information Systems of the experiment.

Monitoring and logging

- Monitoring information system yet have to be developed
  - Will become necessary as detector components become ready
  - Will use various source of data from the subsystem and other databases
  - Data will be transferred in JSON format with variable schema
    - Only few mandatory fields required (source id, time stamp, etc..)
  - Data transfer through the HTTP requests have to be used
  - Time series database should be used as a backend
  - Commonly used solution like Grafana for use for visualization of data
  - Solutions that worked well on the other NICA experiments should be used
- System for logging and bookkeeping can be also common for the NICA experiments

Database of personnel and documents

- About 400 people are currently participating in the SPD project,
  - number of participants is expected to grow close to experiment start
- In order to organize effective cooperation with the shared use of computing and other resources, it is necessary to have IS for
  - handling of a personnel and organizations data
  - support for working groups: membership, access rights,
  - accounting of the contribution (if implemented)
  - generating reports broken down by various parameters
- Procedures for creating, approving and editing related documents
  - Registration and changes of membership in the collaboration
  - Creating and editing lists of groups and privileges
  - Inclusion in the author's lists

Database of publications

- SPD already produced some publication and conference reports
- An IS is necessary for preparing and publishing results
  - Tracking the progress of publications,
  - Organizing the exchange of messages between authors, reviewers and curators
  - Searching through documents
  - Tracking of SPD publications in external information systems.
  - Repports on the number of publications, broken down by authors, topics…
- Organization of presentations and reports at conferences and meetings:
  - Compiling a list of conferences and available reports
  - Organization of call for speakers and selection
  - Acceptance, review and approval of titles, abstracts and slides
  - Tracking the publication of proceedings

Identification and autorisation

- To identify employees, JINR authentication services should be used
  - Providing possibility of access by external employees who do not have an JINR account
  - Introducing group and robot accounts for use in automated tasks
  - A role and privilege management system should be implemented as well as group access policies
- Procedures for including users in groups and revoking membership
  - Information should be provided by the experiment to the JINR IT services
- The resources of the working groups should use a single authentication and authorization system

Databases organization

- IS should use database clusters built on common base of servers and storages for reducing cost and maintenance efforts to serve different type of application workload
  - PostgreSQL currently is being used as a base DBMS solution
  - A high performance, professionally manged common service is necessary
- The nature of the distributed computing generates dynamic workload. Coping with high loads on the database side is essential.
  - Three Tier model should be applied when necessary.
  - Direct access to the databases should be avoided.
  - Application level should be introduced, it should be responsible for the user interface and API, providing caching and asynchronous access

Databases organization

- Databases and applications should be designed aiming for scalability, having in mind long term operation in the varying conditions, with different data flows and request rates.
  - Adaptation to the varying data formats and content should be provided
  - Use of common flexible formats like JSON recommended
- Development and deployment would take quite a long period, some technologies may become obsolete or not available,others may emerge
  - Possibility of transfer to the new platform should be kept in mind
  - Use of module design, container-like solution should be encouraged
- Open source solutions should be preferred

Contribution required

- If you willing to participate in the Database, API or user Interface development, please join
- As it was mentioned before, Information systems should be tailored to the needs of the project and to the nature and amount off data
- We need input both from hardware and analysis groups to create information systems fitting their need
  - If you created some database for your subsystem, please share your experience so it may be implemented elsewhere
  - If you have list of hardware (with parameters) that will be used in your system, contact us so we may adjust database and interface to it
  - The same if you system needs API to one of the information systems

Contribution required

- What we desperately need from the detector subsystems
- More details about detector calibration procedures and constants
  - Details and naming convention for geometry description
  - Input for the database design
  - Detector hardware database (detector elements, cabling etc)
  - Run database
  - Offline DB: Geometry versions, Calib&Align, Magnetic feld, …
- We need all it rather early to have time for proper design, performance tests and tuning