SPD OnLine Filter. Status update

VIII SPD Collaboration Meeting. 07.11.2024 Oleynik Danila.

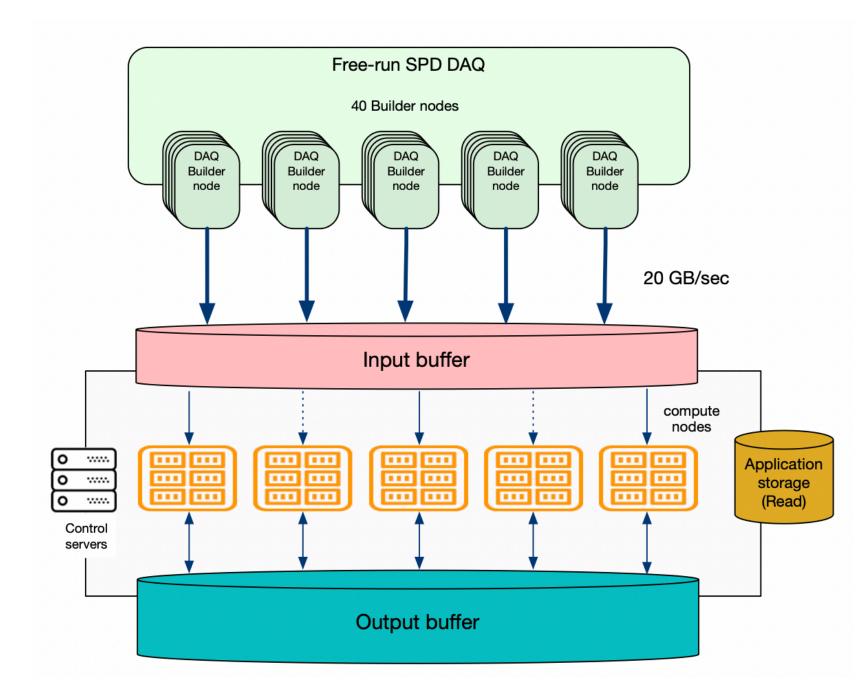
SPD Online filter Reminder :-)

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 working nodes: multi-CPU and hybrid multi CPU + Neural network accelerators (GPU, FPGA etc.)
 - Middleware component: software complex for management of multistep data processing and efficient loading (usage) of computing facility.
 - Applied software: performs informational processing of data

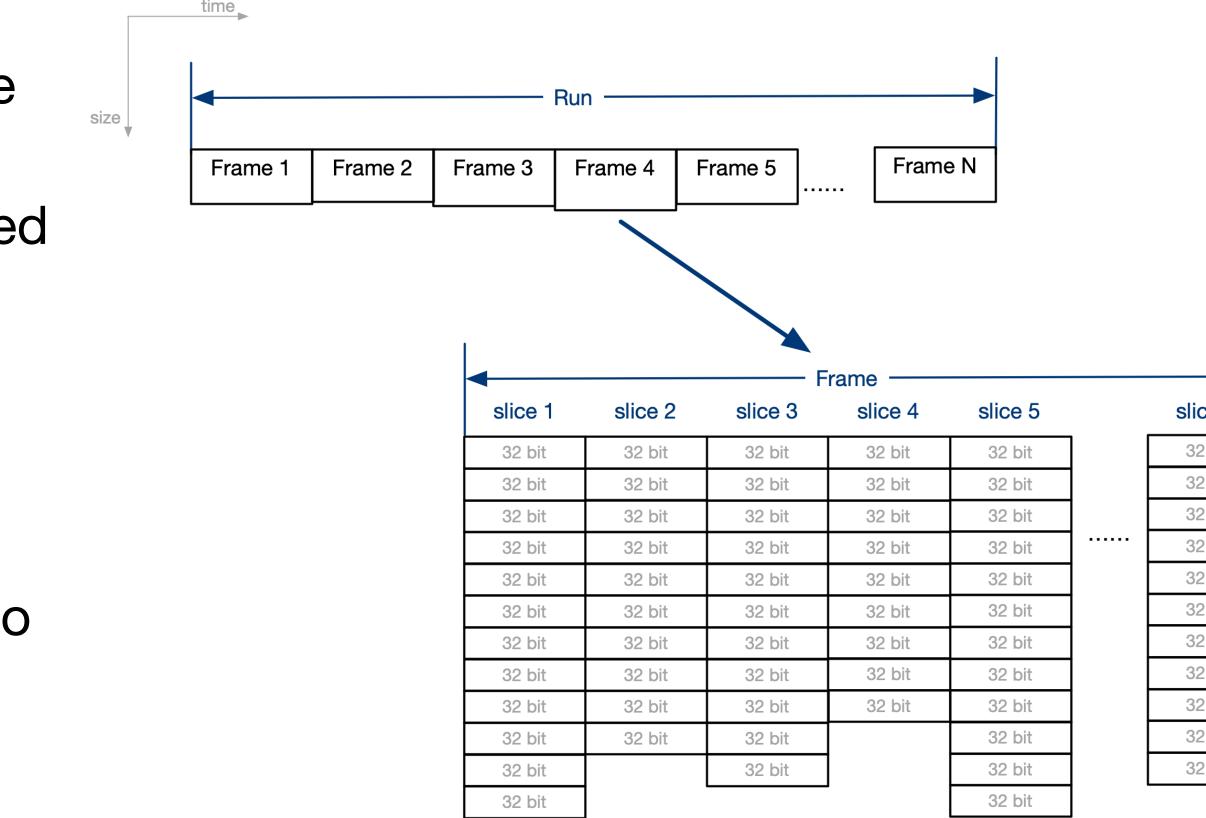




Initial data

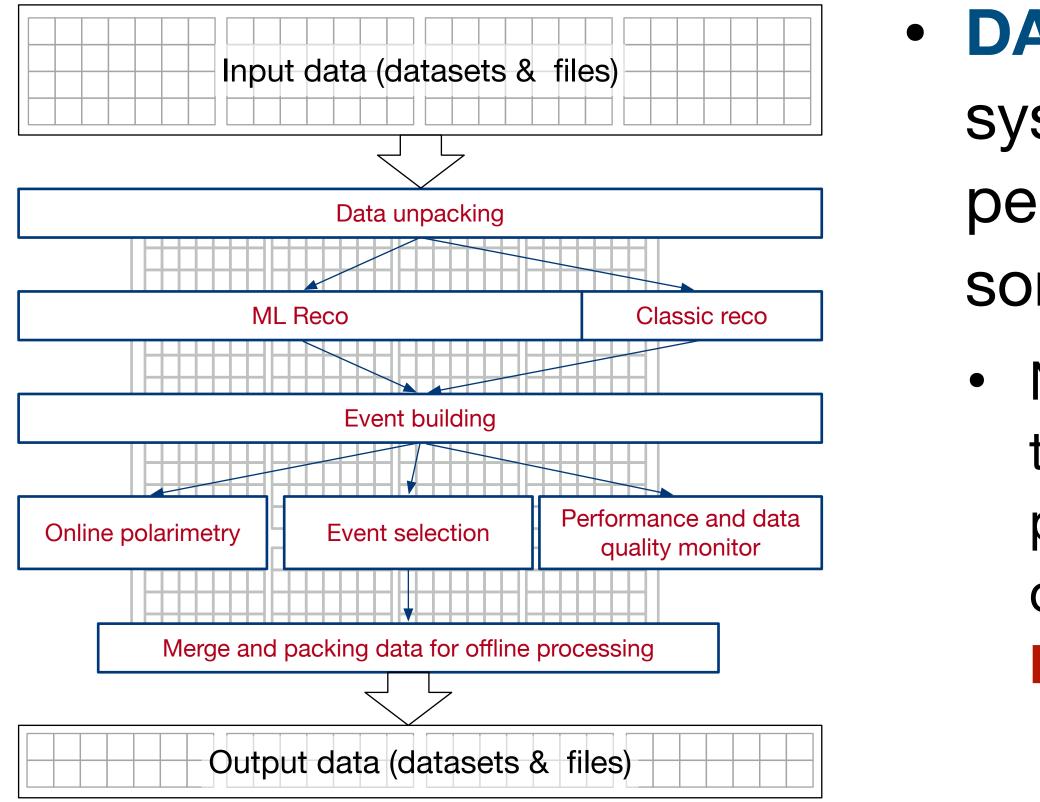
- Free run DAQ, means that the output of the system will not be a dataset of raw events, but a set of signals from detectors organized in time slices
- Primary data unit: time slice (~10 µs)
 Time slices combined in time frames (1-10 sec.)
- Every slice will contain signals from a few to many collisions (events)
- Event building have to unscramble events from a series of time slices





slice N					
32 bit					
32 bit					
32 bit					
32 bit					
32 bit					
32 bit					
32 bit					
32 bit					
32 bit					
32 bit					
32 bit					

Data 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 – 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**



Infrastructure Current situation

- Integration infrastructure deployed (currently)
 - 4 VM for debugging of general services components
 - 1 imitation of storages
 - 1 Dedicated DBMS Server with PostgreSQL 16
 - Rabbit MQ dedicated facility are going to deploy soon
- AlmaLinux 9 base OS. It will work for us next few years, security support till the middle of 2032
- GIT based CI/CD functionality provided by JINR LIT (Cloud infrastructure)



Integration infrastructure deployed on JINR Cloud resources on 6 VM's

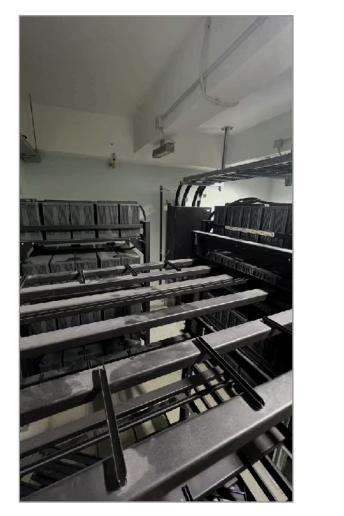
Infrastructure Hardware for prototyping (NICA Computing)

- Building 14 almost ready to host a NICA Computing data center
- In progress purchasing of some compute and storage facility

Compute unit	Storage unit	GPU unit	Control units
	2 CPU * 44 Core (FMA3, AVX512)	2 CPU * 48 Core (FMA3, AVX512)	2 CPU * 44 Core (FMA3, AVX512)
	512 GB RAM	1024 GB RAM	512 GB RAM
	32 * 30,72 TB NVMe - 983 TB	8 GPU (NVIDIA H100)	2 * 4 TB NVMe
Ethernet: 100 Gb/s	InfiniBand: 2 * 200 Gb/s	4 * 4 TB NVMe - 16 TB per node	Ethernet: 100 Gb/s
InfiniBand: 200 Gb/s		InfiniBand: 2 * 200 Gb/s	InfiniBand: 200 Gb/s

- 32 Compute units -> 2816 Cores with ~10GB RAM per core
- 2 Storage units -> **1,8 PB Storage raw volume**
- 1 GPU unit -> 8 NVIDIA H100 GPU







Middleware

Data management;

• Support of data life-cycle and storage usage;

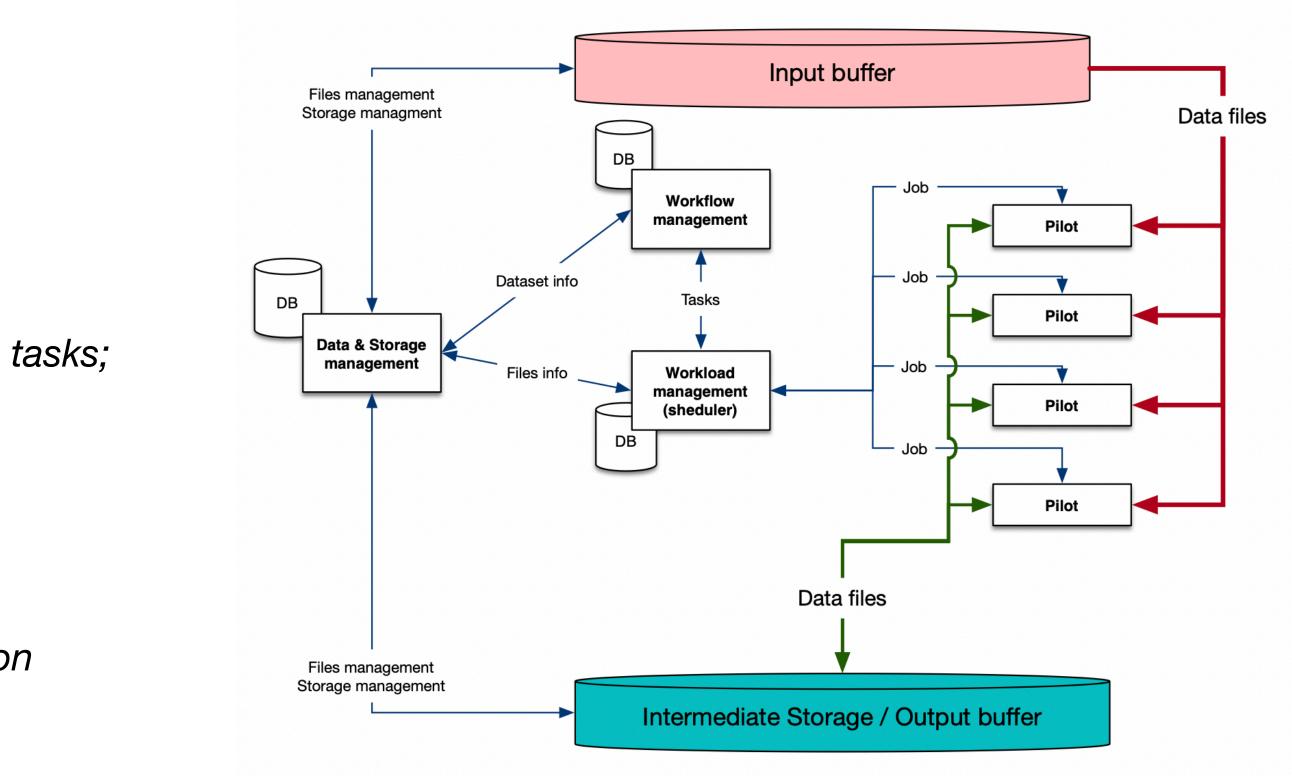
Workflow management;

- 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 \bullet compute nodes;
- Manpower: one PHD students (SPbSU) now hired by LIT, one full time researcher (JINR LIT), two master students (MEPHI) - hired by LIT





Middleware active development, more details in Nikita's report

spit uanfas: ', ', ', ', ', ', ', ', ', ', ', ', ',	<pre>itter-collector-1 INF0: 2024 itter-collector-1 INF0: 2024 itter-collector-1 INF0: 2024- it.code': 0, 'datetime': '2024-12 ', 'size': 26, 'check_sum': 'e3 e': 'file_out.task_test1.job_272 92427ae41e4649b934ca495991b78524 sk_822/job_272/file_out.task_test2 '}, {'id': 978, 'job_id': 272 check_sum': 'e3b0c44298fc1c149aa' 'file_url': '/data/SPD0F-buffers state': 'ready', 'error_code': 0 ile_log.task_test1.job_272.tar.g ics': {'inf0': 'Job metrics', 'd exit_code': 0, 'datetime': '2024 put', 'size': 26, 'check_sum': name': 'file_out.task_test1.job 56b92427ae41e4649b934ca495991b78 /task_822/job_272/file_out.task o': '}, {'id': 978, 'job_id': 27 /task_822/job_272/file_out.task o': '}, {'id': 978, 'job_id': 27 /task_822/job_272/file_out.task itter-collector-1 INF0: 2024- itter-collector-1 INF0: 2024- its' 'file_out.task_test1.job_272 /cask_822/job_272/file_out.task info': ''}, {'id': 978, 'job_id 419, 'check_sum': 'e3b0c44298fc2 22/job_272/file_log.task_test1. ''}], 'metrics': {'info': 'Job code': 0, 'exit_code': 0, 'date ''ype': 'input', 'size': 26, 'c 272, 'file_name': 'file_out.task 149afbf4c8996fb92427ae41e4649b9 -buffers/output/task_822/job_272/file_ 4934ca495991b7852b855', 'state': s/output/task_822/job_272/file_'d e': 0, 'info': ''}], 'metrics' s/output/task_822/job_272/file_'d e': 0, 'info': ''}], 'metrics' s/output/task_822/job_272/file_'d e': 0, 'info': ''], 'metrics'</pre>	-11-05 09:50:15,00 1-05 09:50:15.0620 b0c44298fc1c149aff 2_12', 'file_url' b855', 'state': 'f st1.job_272_28', , 'file_name': 'f fbf4c8996fb92427ac s/output/task_822, 0, 'info': ''}], gz', 'type': 'log cpu_load': 0.3, 'f 4-11-05 09:50:15.0 'e3b0c44298fc1c14 _272_12', 'file_uf 852b855', 'state' _test1.job_272_28 272, 'file_name': 49afbf4c8996fb924 fers/output/task_8 272, 'file_name': 49afbf4c8996fb924 fers/output/task_8 ': 0, 'info': ''} ar.gz', 'type': ' , 'cpu_load': 0.3 -11-05 09:50:16.169 b0c44298fc1c149aff 2_12', 'file_url' b855', 'state': ' _test1.job_272_28 ': 272, 'file_name' 2_12', 'file_url' b855', 'state': ' _test1.job_272_28 ': 272, 'file_name' 2_12', 'file_url' b855', 'state': ' _test1.job_272_28 ': 272, 'file_name' c149afbf4c8996fb9 0F-buffers/output, rror_code': 0, 'if job_272.tar.gz', metrics', 'cpu_lo etime': '2024-11-0 check_sum': 'e3b0 sk_test1.job_272_3 34ca495991b7852b8 2/file_out.task_te ': 978, 'job_id': : 'e3b0c44298fc1c2 rl': '/data/SPD0F 'finished', 'errof log.task_test1.job	54 collector.s 560', 'files_in_ 560', 'files_in_ 560', 'files_in_ 560', 'error_c 'ype': 'output' ile_out.task_tes 560', 'error 'ype': 'output' ile_out.task_tes 560', 'files_ 560', 'files_ 5700', 'files_ 5700', 'files_log_ur 5600', 'size': 800 5700', 'size': 800 5600', 'error 5700', 'size': 800 5700', 'siz	<pre>e41e4649b934ca495 uffers/output/tas ode': 0, 'info': , 'size': 412, 'o t1.job_272_15', ' 5991b7852b855', ' t.task_test1.job_ [{'id': 923, 'jo 'check_sum': 'e3k .2, 'network_load in_url': [{'id': 27ae41e4649b934ca F-buffers/output/ r_code': 0, 'info ut', 'size': 412, test1.job_272_15' a495991b7852b855' _out.task_test1.j l': [{'id': 923, 9, 'check_sum': ' 11.2, 'network_l ess web_log.py: rc.job_manager url': [{'id': 353 e41e4649b934ca495 uffers/output/tas r_code': 0, 'info ut', 'size': 412, sk_test1.job_272_ 34ca495991b7852b855' uffers/output/tas r_code': 0, 'info ut', 'size': 412, sk_test1.job_272_ 34ca495991b7852b8 2/file_out.task_test1.j 2/size': 809, 'check_sum': ' 11.2, 'network_test1.j 2/file_out.task_test1.j 2/file_out.task_test1.j 2/file_out.task_test1.j 3/, 'files_in_ur f4c8996fb92427ae4 '/data/SPD0F-buf inished', 'error_ , 'type': 'output ': 'file_out.task_test2427ae41e4649b934 task_822/job_272/ fo': ''}], 'files type': 'log', 'si ad': 2.2, 'memory</pre>	<pre>job_manager 3, 'job_id': 5991b7852b85 5k_822/job_2 ''}, {'id': theck_sum': file_url': state': 're 272_13', 't b_id': 272, 0c44298fc1c 'state': 're 272_13', 't b_id': 272, 0c44298fc1c 'ask_822/job_id': 2495991b7852 'task_822/job ': ''}, {'i 'check_sum , 'file_url , 'state': job_272_13', 'job_id': 2 e3b0c44298f coad': 11713 211 1 >>> job_manager } job_id': 2 e3b0c44298f coad': 11713 211 1 >>> job_id': 2 e3b0c44298f coad': 11713 22 e3b0c44298f coad': 1172</pre>
spi	itter-collector-1 INF0: 2024	-11-05 09:50:16,1	71 aiohttp.acc	ess web_log.py:	211 1 >>>



.py:27 (I >>> json={'job_id': '272', 'pilot_id': '1', 'info': '', 'state': 'stage_out', 'error_code': 0, ' 272, 'file_name': 'file_in_13', 'file_url': '/data/SPDOF-buffers/input/builder_01/file_in_13', 'type 'state': 'finished', 'error_code': 0, 'info': ''}], 'files_out_url': [{'id': 173, 'job_id': 272, 'file_ out.task_test1.job_272_12', 'type': 'output', 'size': 894, 'check_sum': 'e3b0c44298fc1c149a1 'job_id': 272, 'file_name': 'file_out.task_test1.job_272_28', 'file_url': '/data/SPDOF-buff 'e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855', 'state': 'ready', /data/SPD0F-buffers/output/task_822/job_272/file_out.task_test1.job_272_15', 'type': 'output', : 419 'error_code': 0, 'info': ''}, {'id': 482, 'job_id': 272, 'file_name': 'file_out.task_test1.job_272_13 output', 'size': 99, 'check_sum': 'e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855 'file_name': 'file_log.task_test1.job_272.tar.gz', 'file_url': '/data/SPD0F-buffers/output/task_822/job_272 149afbf4c8996fb92427ae41e4649b934ca495991b7852b855', 'state': 'finished', 'error_code': 0, 'info': ''}], 'me 'io': 12211}}, dict={'job_id': '272', 'pilot_id': '1', 'info': '', 'state': 'stage_out', 'error_code': 0 : 272, 'file_name': 'file_in_13', 'file_url': '/data/SPD0F-buffers/input/builder_01/file_in_13', 'type': state': 'finished', 'error_code': 0, 'info': ''}], 'files_out_url': [{'id': 173, 'job_id': 272, 'fi le_out.task_test1.job_272_12', 'type': 'output', 'size': 894, 'check_sum': 'e3b0c44298fc1c149afbf4c8 'job_id': 272, 'file_name': 'file_out.task_test1.job_272_28', 'file_url': '/data/SPDOF-buffers/out 'e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855', 'state': 'ready', 'error_code': '/data/SPDOF-buffers/output/task_822/job_272/file_out.task_test1.job_272_15', 'type': 'output', 'size': 'error_code': 0, 'info': ''}, {'id': 482, 'job_id': 272, 'file_name': 'file_out.task_test1.job_272 'output', 'size': 99, 'check_sum': 'e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b85 72, 'file_name': 'file_log.task_test1.job_272.tar.gz', 'file_url': '/data/SPD0F-buffers/output/task_822/job c1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855', 'state': 'finished', 'error_code': 0, 'info': ''}], 'io': 12211}}, type=<class 'dict'> 🚬 🕻 🚰 📣 [05/Nov/2024:09:50:15 +0000] "POST /status HTTP/1.1" 200 295 "-" "Python-urllib/3.12" .py:2//) >>> json={'job_id': '272', 'pilot_id': '1', 'info': '', 'state': 'completed', 'error_code': 0, ' 'file_name': 'file_in_13', 'file_url': '/data/SPD0F-buffers/input/builder_01/file_in_13', 'type': 'inp 'state': 'finished', 'error_code': 0, 'info': ''}], 'files_out_url': [{'id': 173, 'job_id': 272, _out.task_test1.job_272_12', 'type': 'output', 'size': 894, 'check_sum': 'e3b0c44298fc1c149afbf4c8996 405, 'job_id': 272, 'file_name': 'file_out.task_test1.job_272_28', 'file_url': '/data/SPDOF-buffers/out 'e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855', 'state': 'finished', 'error_code': 0 url': '/data/SPDOF-buffers/output/task_822/job_272/file_out.task_test1.job_272_15', 'type': 'output', 'size 'error_code': 0, 'info': ''}, {'id': 482, 'job_id': 272, 'file_name': 'file_out.task_test1.j 'type': 'output', 'size': 99, 'check_sum': 'e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7 'file_name': 'file_log.task_test1.job_272.tar.gz', 'file_url': '/data/SPD0F-buffers/output/tasl p0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855', 'state': 'finished', 'error_code': 'info ': 117278374, 'io': 12211}}, dict={'job_id': '272', 'pilot_id': '1', 'info': '', 'state': 353, 'job_id': 272, 'file_name': 'file_in_13', 'file_url': '/data/SPDOF-buffers/input/builder_01/file_in_13 a495991b7852b855', 'state': 'finished', 'error_code': 0, 'info': ''}], 'files_out_url': [{'id': 173, 'job_i /task_822/job_272/file_out.task_test1.job_272_12', 'type': 'output', 'size': 894, 'check_sum': info': ''}, {'id': 405, 'job_id': 272, 'file_name': 'file_out.task_test1.job_272_28', 'file_url': '/data/SPI 412, 'check_sum': 'e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855', 'state': 'finished', 272_15', 'file_url': '/data/SPDOF-buffers/output/task_822/job_272/file_out.task_test1.job_272_15', 'type' 852b855', 'state': 'finished', 'error_code': 0, 'info': ''}, {'id': 482, 'job_id': 272, 'file_name': 'file_ou ask_test1.job_272_13', 'type': 'output', 'size': 99, 'check_sum': 'e3b0c44298fc1c149afbf4c8996fb92427ae41e464 [{'id': 923, 'job_id': 272, 'file_name': 'file_log.task_test1.job_272.tar.gz', 'file_url': '/data/SPDOF-buff 'check_sum': 'e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855', 'state': 'finished', 'error_ 2, 'metwork_load': 117278374, 'io': 12211}}, type=<class 'dict'> [05/Nov/2024:09:50:16 +0000] "POST /status HTTP/1.1" 200 295 "-" "Python-urllib/3.12"

Applied software for Online Filter

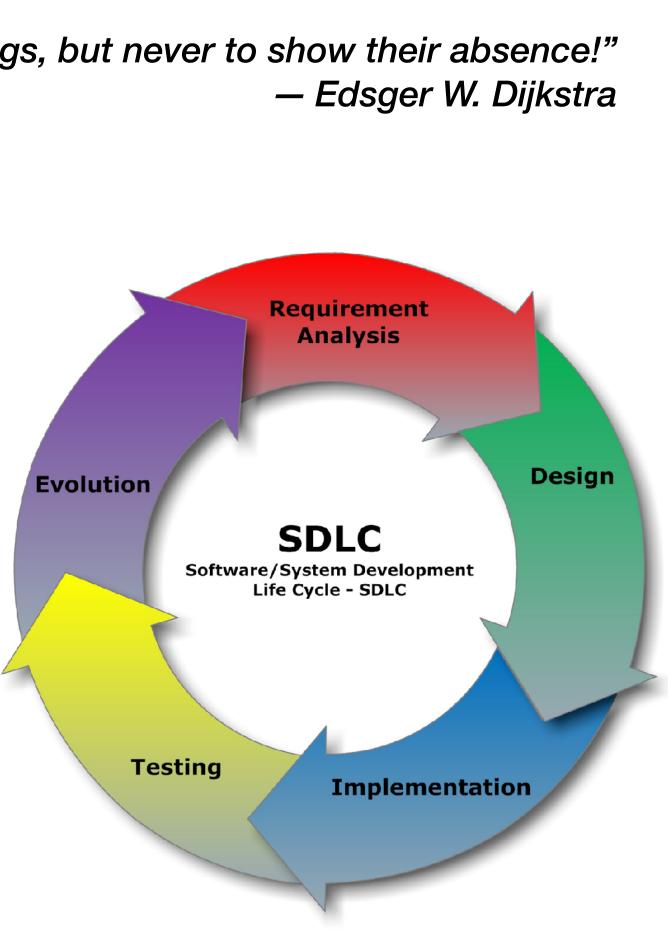
- Sampo framework will be used for realisation of applied software for SPD Online filter
 - Initial functional and non-functional requirements should be collected in next couple of months
- ML algorithms was presented at CHEP (Krakow) and MMCP (Yerevan) conferences
 - Integration of ML tools with Sampo will became one of the topics



A bit boring about (any) software

"Program testing can be used to show the presence of bugs, but never to show their absence!" — Edsger W. Dijkstra

Software errors are flaws that result unexpected outcome from a computer program or system or cause it to act in ways that were not intended. They are any mismatches between the software and its requirement specification. How software errors are illuminated? Some part of them by testing, others by usage experience. The only way to fix error(s) are launching next iterate of development cycle



A bit boring about ML usage?

- A lot of similarity with software life cycle, is not it?
- In most cases should work, but mismatches with expected result is unpredictable
 - so detection of mismatching of results with expected should trigger process of retraining of NN
 - except the situation with poor quality of input data, but in this case we should face similar issues with 'traditional' methods.
 - Deep anomaly detection refers to any methods that exploit deep learning techniques to learn feature representations or anomaly scores for anomaly detection.
- In any case requirements for any ML tools should be **collected and formalised** as much as possible (as for any software)
 - As only we will reach enough formalisation they will be applicable for developing of proper algorithm

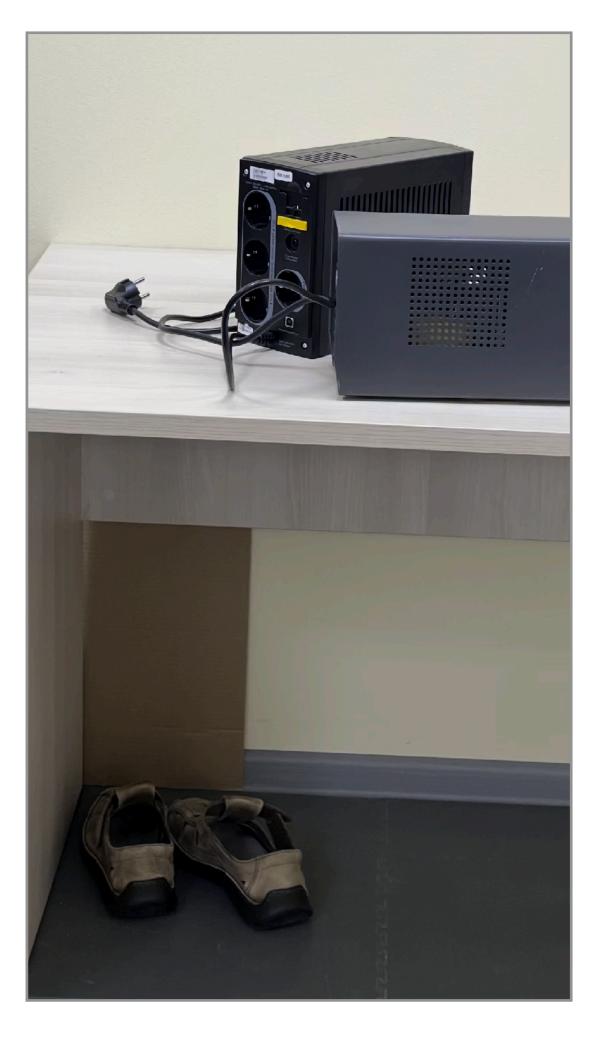
DAQ & Online filter testbed in MLIT

- Testbed zone is ready
- Some of habitants (a couple of summer students Andrey and Dmitrii) appeared this summer











Thank you!