

JOINT INSTITUTE FOR NUCLEAR RESEARCH

Meshcheryakov Laboratory of Information Technologies

**FINAL REPORT ON THE**

**START PROGRAMME**

*PanDa Pilot3 for SPD*

**Supervisor:**

Artem Sh. Petrosyan

**Student:**

Uriy Basharimov,
Sukhoi State Technical
University Of Gomel

**Participation period:**

July 8 – August 15,

Summer Session 2024

Dubna, 2024

**TABLE OF CONTENTS**

ABSTRACT 3

INTRODUCTION 4

NICA SPD 5

PANDA PILOT 7

RESULT 9

SPD PILOT FUTURE 10

BIBLIOGRAPHY 12

ACKNOWLEDGMENTS 13

**АBSTRACT**

The PanDa (Production and Distributed Analysis) pilot system was developed for job execution and data movement in the ATLAS experiment at the LHC. It utilizes pilot jobs sent to computing sites to establish a controlled execution environment for handling production and user analysis jobs. The pilots validate the worker node environment, configure it appropriately, and pull subsequent real payload jobs through prioritized queues. This decouples the workflow management from the heterogeneous grid infrastructures, improving reliability and scalability.

The pilot concept proved successful in meeting ATLAS's intensive computing needs, handling millions of concurrent jobs across over 100 sites. It streamlined operations by transparently relocating jobs upon failures, leveraging late binding of resources more efficiently than traditional batch systems.

The PanDa pilot system's design overcame many challenges in the complex ATLAS computing model, enabling the experiment to exploit distributed resources while insulating the workflow from site heterogeneities.

**INTRODUCTION**

Efficiently utilizing distributed computing resources across heterogeneous sites and infrastructures is a major challenge in large-scale scientific data processing. Traditional batch systems often suffer from inefficiencies, loss of resources due to job volatility, and inability to adapt to diverse site conditions. The pilot job concept provides an elegant solution to this problem by establishing a controlled execution environment at computing sites before pulling and executing the actual payload jobs.

The core idea behind pilot workflows (fig. 1) is the late-binding of pilot processes to real computational work. Pilot jobs are lightweight processes that are first submitted to diverse computing resources like grid sites, clouds or HPC clusters by a central manager system. Upon starting at a site, the pilot validates resource capabilities, configures the software environment, and sets up input data before signaling readiness to the manager. Only then does the pilot initiate an internal pull mechanism to retrieve and execute prioritized payloads sent through central job queues.



Figure 1 – Pilot workflow

This approach offers several key advantages over traditional systems. It facilitates high throughput computing by automatically redirecting jobs from unavailable sites. Resource loss from job volatility is reduced as pilots can re-submit or replicate failed jobs. The workflow is isolated from site heterogeneities as pilots handle environment configuration. Just-in-time data and software distribution to sites is enabled through the pilots. Overall, the pilot abstraction layer allows efficient sharing of resources and dynamic adaptation to site conditions.

Pilot job frameworks like PanDA, HTCondor, DIRAC have been widely adopted in large scientific computing projects due to their ability to effectively harness distributed resources for data-intensive workloads across grid, cloud and HPC infrastructures. As computational demands continue growing exponentially, intelligent pilot factories, scheduling algorithms and monitoring capabilities are becoming crucial for optimal utilization of heterogeneous distributed resources. Many findings indicate the pilot paradigm is well-suited to meet emerging computing challenges in data-intensive scientific domains.

**NICA SPD**

The Nuclotron-based Ion Collider fAcility (NICA) is a new accelerator complex under construction at the Joint Institute for Nuclear Research (JINR) in Dubna, Russia. One of the major experiments at NICA will be the Spin Physics Detector (SPD), aimed at investigating the spin structure of nucleons and studying the properties of dense baryonic matter.

The SPD is a multi-purpose detector system designed to register charged particles, electromagnetic probes, and nuclear fragments produced in collisions of polarized proton and deuteron beams with momentum up to 27 GeV/c per nucleon. The key physics goals are to study the spin structure of nucleons using polarized beam and target, search for manifestations of the deconfinement phase transition and critical phenomena, and investigate properties of dense baryonic matter.

The main components of the SPD include:

- superconducting solenoid magnet providing a maximum field of 0.6 T;

- an electromagnetic calorimeter system for energy measurements;

- a time-of-flight system for particle identification;

- a focusing aerogel RICH detector for charged hadron identification;

- a straw tube tracker for momentum measurements.

The SPD is optimized to operate at average interaction rates up to 7 kHz for inelastic events. It will record minimum bias, as well as selected rare probes using a high-performance trigger and data acquisition system.

This state-of-the-art detector will provide unprecedented statistics and kinematic range to explore the spin structure of nucleons and the properties of dense nuclear matter produced in heavy-ion collisions at NICA energies. The broad physics program spans both fundamental and applied research topics at the frontiers of nuclear physics.

The Spin Physics Detector at NICA has a broad physics program with several key expected outcomes.

Spin structure of nucleons. The SPD will use polarized proton and deuteron beams to carry out extensive studies on the spin structure of nucleons. Precise measurements of the gluon contribution to the nucleon spin and investigations of the orbital angular momentum contributions are expected to shed new light on how the spin of nucleons arises from the spin and orbital motions of its constituent quarks and gluons.

Deconfinement and critical phenomena. Collisions of heavy ions at NICA energies could create conditions of high baryon density and moderate temperatures, allowing exploration of the QCD phase diagram in the region of high net baryon densities. The SPD data may reveal signs of a deconfinement phase transition and critical phenomena associated with the restoration of chiral symmetry.

Properties of dense baryonic matter. The SPD will probe the equation of state and transport properties (e.g. viscosity) of the dense nuclear matter created in heavy-ion collisions. Fluctuation and correlation measurements could provide insights into the degrees of freedom that govern the dynamics and characterize different phases of baryonic matter.

Nucleon structure at short distances. Measurements of hard probes like jets, high-pT hadrons and direct photons can map out the modification of nucleon structure at short distances inside the nuclear environment, testing phenomena like color transparency and hadronization processes.

Hyperon physics and strangelets. Spectroscopy of multi-strange hyperon resonances and searches for strange metastable exotic objects like strangelets are among the potential discoveries enabled by the SPD's particle identification capabilities.

Overall, the SPD program aims to substantially advance knowledge about fundamental questions of nucleon spin structure, non-perturbative QCD dynamics, and the properties of highly dense baryonic matter – complementing the efforts at other facilities worldwide.

The NICA SPD has a rich physics program with several key goals across different frontiers of nuclear and particle physics research.

Nucleon spin structure and parton dynamics – Precisely measure the gluon contribution to the nucleon spin. Study the orbital angular momentum contributions to nucleon spin. Investigate the spin-dependent parton distribution functions. Explore the dynamics of partons at high gluon densities.

Phase transitions and critical phenomena – Search for signatures of the deconfinement phase transition. Study the restoration of chiral symmetry at high baryon densities. Probe the QCD phase diagram in the high net-baryon density region. Investigate the possible critical point and phase boundary.

Properties of dense baryonic matter – Determine the equation of state of dense nuclear matter. Measure transport properties like shear viscosity over entropy ratio. Study the relevant degrees of freedom in different density/temperature regimes. Explore the hadron production mechanisms and space-time evolution.

Nucleon structure modification and hadronization – Investigate color transparency phenomena and small-x physics. Study jet quenching and high-pT hadron suppression. Explore hadronization processes in cold nuclear matter.

Strangeness and exotic hadron physics – Spectroscopy of multi-strange hyperon resonances. Search for metastable exotic objects like strangelets. Measure hyperon polarization and study strange particle production.

With its capability to create and probe high baryon density matter, NICA provides a unique opportunity complementary to studying the quark-gluon plasma formed at high temperatures at RHIC and LHC. The SPD's extensive particle identification and wide kinematic coverage will enable precision measurements to address these fundamental questions in strong interaction physics.

**PANDA PILOT**

The PanDA Pilot is a transient agent that executes jobs on worker nodes. It retrieves job information from the PanDA server, prepares the execution environment, runs the job payload, and manages the job lifecycle on the worker node. Throughout its operation, the Pilot periodically reports various metrics and status updates back to the PanDA server over HTTPS. This allows the central system to monitor job progress and resource utilization in real-time.

The Pilot is responsible for handling many low-level details of job execution (fig. 2), such as setting up software environments, staging input data, executing the actual scientific application, collecting output data, and cleaning up afterwards. It acts as an intermediary between the central PanDA services and the local computing resources, abstracting away differences between diverse computing environments.



Figure 2 – Handling of job

By running as a lightweight process on the worker node, the Pilot can closely monitor resource usage, detect problems, and take appropriate actions like terminating stuck jobs. This improves overall reliability and efficiency of the distributed computing system. The Pilot's design allows it to operate in a variety of computing environments, from traditional grid sites to clouds and high-performance computing centers.

The PanDA Pilot employs a modular architecture designed for flexibility and extensibility. At its core is the Pilot module, which orchestrates the overall job execution process. This core interacts with various subcomponents to handle specific aspects of job management.

The Payload module is responsible for setting up the runtime environment and executing the actual scientific application. It handles software installation, environment configuration, and invocation of the payload process.

For data handling, the Data module manages input staging and output transfer. It interfaces with different storage systems and protocols to efficiently move data to and from the worker node.

The Monitoring module collects real-time metrics about job progress, resource utilization, and system health. These metrics are periodically sent back to the central PanDA server.

To interface with diverse computing resources, the Pilot includes a Resource module. This component abstracts away differences between various batch systems, cloud providers, and other execution environments.

Error handling and recovery are managed by the Recovery module, which implements strategies to detect and respond to various failure scenarios.

The entire Pilot is designed to be lightweight and self-contained, typically packaged as a Python application that can be easily deployed across heterogeneous computing resources. Its modular design allows for easy customization and extension to support new resource types or experiment-specific requirements.

The PanDA Pilot architecture offers numerous advantages in distributed computing environments. Its modular design enables easy customization and extension, allowing experiments to tailor the Pilot to their specific needs while maintaining a common core. This flexibility supports a wide range of computing resources, from traditional grid sites to cloud platforms and high-performance computing centers.

By executing jobs locally on worker nodes, the Pilot provides fine-grained control over the job lifecycle. This allows for real-time monitoring of resource usage, early detection of problems, and intelligent decision-making based on local conditions. The Pilot can take autonomous actions like terminating stuck jobs or adjusting resource allocation, improving overall system reliability and efficiency.

The Pilot's ability to abstract away differences between diverse computing environments simplifies the central workload management system. It presents a uniform interface to the PanDA server, hiding the complexities of various batch systems, storage solutions, and site-specific configurations. This abstraction layer facilitates the integration of new resource types and enables seamless workload distribution across heterogeneous resources.

Real-time communication between the Pilot and central services enables dynamic workload balancing and responsive system management. The continuous stream of status updates and metrics allows for accurate tracking of job progress and resource utilization across the entire distributed computing infrastructure.

The lightweight nature of the Pilot allows for efficient resource utilization, minimizing overhead on worker nodes. Its self-contained design simplifies deployment and reduces dependencies on site-specific software stacks. This portability is crucial for operating in diverse and evolving computing ecosystems.

Overall, the PanDA Pilot architecture enhances the scalability, reliability, and flexibility of distributed computing operations, enabling efficient processing of massive scientific datasets across global computing resources.

The integration process for new resource types in the PanDA system is designed to be flexible and adaptable. It typically begins with an assessment of the new resource's characteristics, including its batch system, data access methods, and any unique constraints or capabilities.

The process often involves extending or creating new plugins within the Harvester component. Harvester is responsible for submitting pilots to computing resources, and its plugin-based architecture allows for easy addition of support for new resource types. Developers create a new plugin that implements the specific interfaces required for job submission, status checking, and resource management for the new platform.

Modifications to the Pilot may also be necessary to support resource-specific requirements. This could include changes to how the Pilot interacts with the local batch system, manages data, or configures the runtime environment. The modular design of the Pilot facilitates these adaptations without affecting its core functionality.

The PanDA server might require updates to recognize and properly manage the new resource type. This could involve adding new attributes to the queue definitions or modifying job brokerage algorithms to effectively utilize the new resources.

Integration also involves setting up appropriate authentication and authorization mechanisms. This ensures that PanDA can securely access and manage workloads on the new resource type.

Testing is a crucial part of the integration process. This includes verifying basic functionality, such as job submission and execution, as well as more complex scenarios like data handling and error recovery. Performance benchmarking is often conducted to optimize resource utilization.

Throughout the integration, close collaboration with resource providers and stakeholders is maintained to address any site-specific requirements or limitations. Documentation is updated to reflect the new capabilities and any special considerations for using the new resource type.

Finally, the integration is typically rolled out gradually, starting with a pilot phase before moving to full production. This phased approach allows for careful monitoring and fine-tuning of the system to ensure smooth operation at scale.

**RESULT**

In my work with the PanDA (Production and Distributed Analysis) pilot system, I have made several important contributions to enhance its functionality and adapt it for the needs of the NICA SPD experiment. First, I gained a deep understanding of the intricate structure and components of the PanDA pilot codebase. This involved studying the pilot workflow architecture, the various pilot components that handle job payload execution, data and software management, and the interfaces for communication with the PanDA server.

With this knowledge, I undertook the task of updating the pilot code to the latest available version. This ensured the pilot system stays current with the latest features, bug fixes and optimizations from the PanDA development team. Keeping up with the evolving codebase is crucial for utilizing modern capabilities and maintaining stable operation.

A key part of my work focused on integrating support for the NICA SPD experiment into the PanDA pilot system. This involved modifying the pilot to properly handle the specific computing workflows, data formats and software environments required by the SPD workloads. Customizing the pilot for a new experiment is vital to leverage the PanDA distributed computing infrastructure effectively. Furthermore, I implemented enhancements to improve the monitoring and debugging capabilities of the pilot. Specifically, I added functionality to search for and identify specific error patterns in the stdout logs generated by the pilot during payload execution. This facilitates quicker detection, diagnosis and resolution of errors, leading to higher operational efficiency. I also wrote some documentation summarizing the new SPD-specific pilot capabilities and configuration procedures.

During the integration of the PanDA pilot system with the NICA SPD experiment, I encountered several key challenges. One major hurdle was ensuring the pilot's software environment was fully compatible with the specific software stack required by SPD workloads. The SPD computing model relies on a customized set of libraries, frameworks and utilities, making the pilot capable of deploying and interfacing with this software environment correctly at remote sites crucial but non-trivial. Modifying the pilot to properly handle the SPD experiment's unique data formats for its detector outputs and simulated data products during payload staging, input/output marshalling and data transfers was a significant undertaking. Coordination with the SPD computing team was essential to understand the intricacies.

**SPD PILOT FUTURE**

Here are, in my opinion, the next steps to further develop the PanDA pilot system for the SPD experiment.

With the successful integration of core SPD workflows and environment into the PanDA pilot system, the next phase will focus on scaling up operations and enhancing intelligent workflow management capabilities. A key priority is optimizing the pilot job scheduling and workload distribution logic to achieve maximum throughput across global computing sites based on data locality, resource availability and job characteristics. Developing advanced scheduling algorithms that prioritize SMB (Single Mixture Boost) and production chain dependencies while minimizing data movements will be crucial. Furthermore, the error monitoring infrastructure will be expanded with automatic classification methods to identify root causes and facilitate self-healing procedures for prevalent workflow errors.

To support the projected data volumes from the SPD detector, the pilots will be upgraded to enable efficient handling of data-intensive workloads on HPCs and commercial clouds in addition to grid resources. This includes optimizations for high-performance parallel I/O, intelligent data caching strategies, and exploring adoption of serverless computing paradigms. Leveraging machine learning models to predicting resource consumption and make proactive decisions is another area of interest. Integrating the pilot system with modern monitoring tools and dashboards will provide unified operational insights across distributed sites.

From the software development perspective, the pilot codebase will undergo a refactoring process to improve maintainability and modularize components for easier customization across experiments. Standardizing workflow description languages and enhancing interfaces with workflow management systems will simplify usage. Simultaneously, hardening the pilot for enhanced security, data integrity validation and strict endorsement of computing policies will be prioritized. As the NICA/SPD community grows rapidly, scalable documentation and training materials will be developed for onboarding new pilot operators and site administrators. Ultimately, these collective development efforts aim to make the PanDA pilot system a robust, intelligent execution engine capable of harnessing distributed resources worldwide for SPD's data challenges.

**BIBLIOGRAPHY**

Maeno T. et al. PanDA: Production and Distributed Analysis System //Computing and Software for Big Science. – 2024. – Т. 8. – №. 1. – С. 4.

Abazov V. et al. Technical Design Report of the Spin Physics Detector at NICA //arXiv preprint arXiv:2404.08317. – 2024.

PanDa Pilot Wiki : сайт. GitHub. URL: https://github.com/PanDAWMS/pilot3/wiki

**ACKNOWLEDGMENTS**

The author expresses gratitude to Artem Petrosyan and Danila Oleynik for providing comprehensive guidance and moral support in the process of work, as well as for technical support and valuable recommendations. I am also grateful to the organizers of START for the opportunity to complete the program.