DESIGN AND DEVELOPMENT OF APPLICATION SOFTWARE FOR THE MPD DISTRIBUTED COMPUTING INFRASTRUCTURE

A.A. Moshkin[1] [0000-0002-5038-5274], **I.S. Pelevanyuk**[1,a] [0000-0002-4353-493X], **O.V. Rogachevskiy**[1] [0000-0002-5038-5274]

[1] *Joint Institute for Nuclear Research, 6 Joliot-Curie St., Dubna, Moscow Region, Russia, 141980*

E-mail: [a] pelevanyuk@jinr.ru

The Multi-Purpose Detector collaboration began using distributed computing for centralized Monte-Carlo generation in mid-2019. The DIRAC Interware is used as a platform for the integration of heterogeneous distributed computing resources. Since then, workflows of job submission, data transfer, and storage have been designed, tested, and successfully applied. Moreover, the growth of interest in access to the computing system from users is observed. One way to provide such access for the users is to allow them to directly submit jobs to DIRAC. However, direct access to the resources imposes a high responsibility on users and must be restricted. For this reason, another approach was chosen, i.e. to design and develop a dedicated application that collects requirements from users and starts the required number of jobs. Such an approach entails additional effort: elaboration of requirements, application design and development. Nevertheless, it allows for greater control over the workload submitted by other users, reducing possible failures and the inefficient usage of resources.

Keywords: data processing, distributed computing, GRID applications

Andrey Moshkin, Igor Pelevanyuk, Oleg Rogachevskiy

1. Introduction

The Multi-Purpose Detector (MPD) is the first experiment at the NICA complex. The MPD apparatus has been designed as a 4π spectrometer capable of detecting charged hadrons, electrons, and photons in heavy-ion collisions at high luminosity in the energy range of the NICA collider. To reach this goal, the detector will comprise a precise 3D tracking system and a high-performance particle identification (PID) system based on time-of-flight measurements and calorimetry [1].

At present, the NICA complex and the MPD experimental setup are under construction. However, there is a need to generate hundreds of millions of events in order to develop algorithms for recognizing and reconstructing tracks of elementary particles and working out algorithms for physical data analysis. To solve these problems and then analyze real data from the MPD experiment, DIRAC is used.

2. Use of the DIRAC Interware for MPD distributed computing

For efficient data processing of the MPD experiment, a heterogeneous, geographically distributed computing environment is currently being created on top of the DIRAC Interware [2]. The DIRAC Interware is an open-source development platform for the integration of heterogeneous computing and storage resources. It was originally developed for the LHCb computing infrastructure, but was later released as a general-purpose solution for scientific groups. Right now, it is used by many experiments in high-energy physics, particle physics, and astronomy: LHCb, Belle-II, BES-III, CTA, CLIC, ILC. The purpose of DIRAC is to provide access to various computing and storage resources through standard interfaces, namely, web, command line, API, and REST. Another purpose is to provide a set of standard tools for workload management, data management, accounting, workflow management, user management, etc.

The service based on the DIRAC platform was deployed and configured at the Joint Institute for Nuclear Research in 2016. Various tests were performed to estimate its possibilities and performance [3]. By 2019, standard workflow tests related to general Monte-Carlo generation were successfully carried out, which proved the possibility of using DIRAC for real scientific computation. At that time, MPD started to have a necessity for massive computing to perform centralized Monte-Carlo generation for the needs of scientific groups. At first, only Tier1 and Tier2 were used for that work, but later the "Govorun" supercomputer and the VBLHEP cluster were integrated into the system. The JINR EOS storage system was integrated and accessed via root protocol. Authentication is based on x509 certificates, and authorization is regulated by both DIRAC and VOMS.

At the moment, the heterogeneous, geographically distributed computing environment includes the Tier1 and Tier2 grid sites, the "Govorun" supercomputer, the cloud component of the MICC JINR, the computing clusters of VBLHEP JINR and UNAM Mexico [4]. The extensive use of different distributed resources made it possible to successfully complete more than 800 thousand computing jobs. Each job was running approximately 6-7 hours. The total amount of computing work is around 4 MHS06 days. Request for all these jobs came from four MPD Physics Working Groups. They form requests with all the information relevant for data production in a special information system. The production manager uses this information to form job descriptions and submit them to DIRAC. Each job consists of two parts: Job Description information and Shell script that is executed on the worknode.

The Job Description contains information about the resource on which the job should run, what executable will be on the worknode (in most cases for MPD jobs it is Shell: /usr/bin/sh), what arguments should be passed to the executable (it should be a Shell script name), which files should be passed with the job. The Shell script comprises initial data download, configuration, execution of physics software, and result data upload. In most cases, the DIRAC API is used to generate and submit thousands of similar jobs with a different index number. This process requires special expertise related to physics software, the use of DIRAC, and the features of different computing resources.

Right now, there are three major ways to enable users to use distributed computing resources. The first is to perform extensive training for a user and gradually expand his access to different

resources. It is expensive in terms of time both for DIRAC specialists who teach and users who learn. Nevertheless, when the training is complete, the user becomes very flexible in terms of job definition and workflow design. Although, it is required for users to keep up to date with the changing and evolving infrastructure.

The second way is to find a DIRAC specialist to submit physics jobs on behalf of scientists. Users explain the task to the specialist, and the specialist designs and submits jobs to resources. It is less time consuming for users and DIRAC specialists. However, the specialist in DIRAC will need to become more proficient in applied software. This approach has been successfully used by MPD and proved to be effective.

The third way is to develop a dedicated DIRAC application to provide an interface that is defined in terms of user tasks (required software, type of job, energy, number of events needed). It is highly convenient for users. Nevertheless, it requires much more effort from DIRAC development specialists, and it will further require the support of the developed application.

The third approach was chosen to provide physicists with the possibility to use the distributed infrastructure. A special dedicated application was designed and developed.

3. Design and development of the application for MPD users

The DIRAC Interware provides a convenient web interface suitable for small workload submission, job monitoring, accounting check, and some other activities. The frontend is built with the ExtJS framework. Basically, a web interface is a set of different applications related to different activities. Each application requires two files with the code: one with the JavaScript code related to visualization and web logic, another with the Python code working as a backend and responsible for getting data from the database or various DIRAC services. It is possible to create additional custom applications just by adding new modules to the corresponding frontend and backend code directories.

Name of parameter	Type of parameter	Possible values
InputTemplate	string	urqmd-BiBi-09.2GeV-mb-eos0-500
Generator	enum	UrQMD, SMASH, DCQGSM, etc
EventsNumber	int	1000000
Beam	enum	Au, Bi, Ag, p, C, Pb
Target	enum	Au, Bi, Ag, p, C, Pb
Centrality	string	mb
RecMod	string	dst-BiBi-09.2GeV-mb-eos0-500ev
Energy	double	12.2
InputExtention	string	f14
InputSandbox	file	Several uploaded files

Table 1. Input parameters for the MPD application

The creation of a custom DIRAC application that will be responsible for production jobs submission should provide easier access to computing resources with a lower risk of errors. This can potentially bring new users to the system and improve resource utilization. Users will not need to learn all the details about DIRAC, its commands, and API. It was decided to design an application for MPD Monte-Carlo jobs.

The first step is the design of the future application. At this step, it is necessary to define the schema of the future application, the set of fields that will form the input and the set of fields that will form the output. The best way to do it is to analyze Shell scripts of previously submitted jobs. In our case, the main task of the application is to submit thousands of jobs with the same physics parameters

and different index parameters for each job. There should be no output other than the result of massive job submissions. The list of the physics parameters is shown in Table 1.

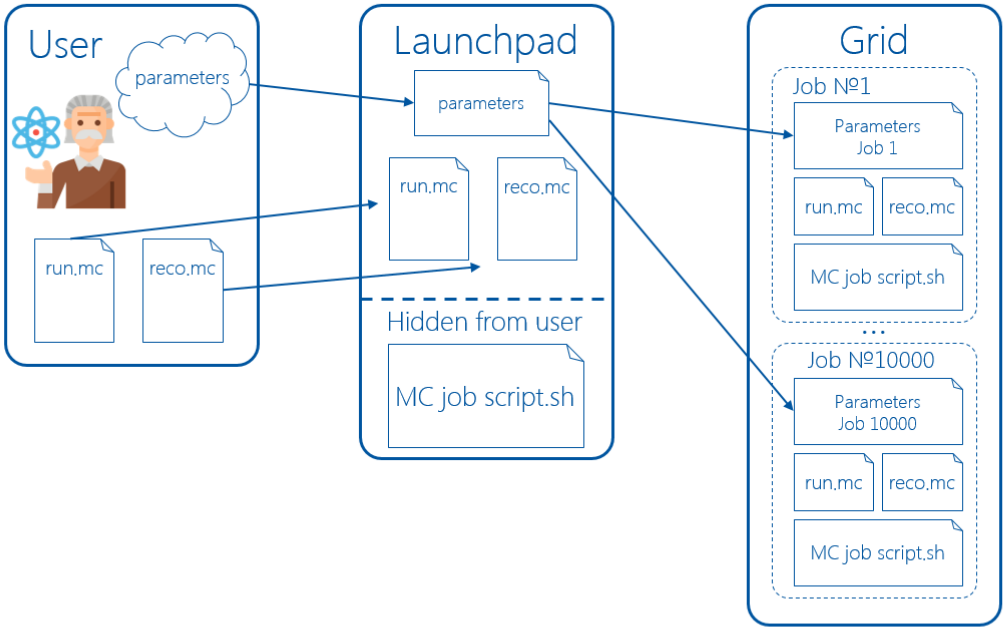

Figure 1. Schema of the MPD application for Monte-Carlo production

The user should open a dedicated application, fill in all the physics parameters, define the number of collision events to be generated, and attach two files with the root code: run.mc and reco.mc. These files are written by physicists and contain the code related to the process of generation and reconstruction correspondingly. On the server side, a special service receives this user request and forms a temporary file with the user's parameters. The file is passed with each job since all jobs should use the same physics parameters. Job Description parameters are set by the service itself, they are the same for all jobs in one pack. To allow it, the Shell script was taken from previous productions and modified to use the physics configuration file. The variable index of each job is passed separately as an additional argument to the Shell script. The required number of jobs is then submitted to the DIRAC Job Queue. The application usage schema is demonstrated in Figure 1.

During the development stage, the major part of the code was taken from the standard DIRAC JobLaunchpad application. Several major changes were made to the developed application. The MPDJobLaunchpad application first checks that the user who opens it belongs to the MPD group in DIRAC. Only MPD users are allowed to submit jobs via this interface. Custom input fields were put on the interface and configured to accept valid parameter values. Most of the logic was put on the server-side code. It reads user parameters, calculates the number of jobs to be submitted, and forms parameters for all submitted jobs. The Shell script for execution now exists on the server. It was changed to accept user parameters and use them during its execution.

4. Conclusion

The developed application allows ordinary users, who are not specialists in the DIRAC Interware, to run massive Monte-Carlo productions. The web interface (Fig. 2) is simple to understand without additional training. It is certainly possible to design and develop new applications if needed. However, the main requirement for using this approach is that only the parameters should be changed, and not the scientific processing schema. Otherwise, frequent changes will be required, and processing will shift to more flexible approaches that imply the help of a DIRAC specialist. The second issue is that the proposed application will not work if scientific software is unstable. Frequent crashes can complicate the process of workload execution, and a lot of additional work from users to resubmit jobs will be required. Although it is possible to improve the developed application in order to provide a resubmission mode, in which it will check for failed jobs and resubmit them.

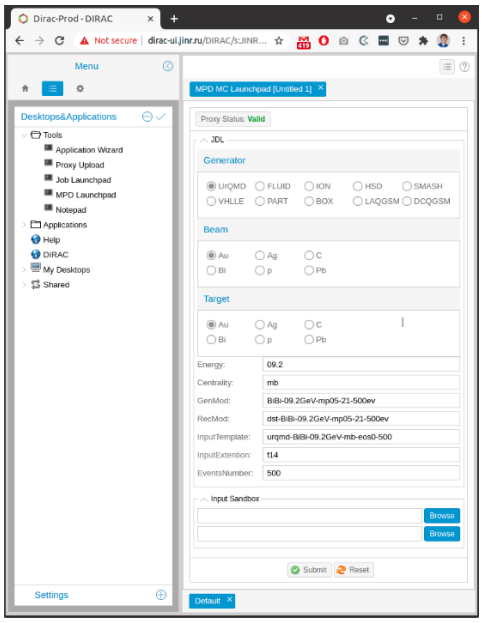

Figure 2. MPD application for Monte-Carlo production on the DIRAC web interface

The main advantage of the proposed approach is that it simplifies the process of submitting jobs to the DIRAC system. It allows us to provide distributed resources to a wider range of users. Another advantage is that it helps to eliminate user errors in execution scripts (not scientific software) and bring more order in result file placement and metadata assignment. If there are no changes in the process of Monte-Carlo generation over time, the support of the application is not required since its work is stable.

The development of dedicated applications for different user groups is a viable option for administrators of DIRAC instances in different organizations. It is not too difficult to design and develop this kind of application, although it requires some expertise in DIRAC development. Developed applications can be used in parallel with other approaches to submitting jobs during the testing stage and, in the case of success, can be proposed to other users.

Acknowledgments

The software described in this work was created with the support of the RFBR special grant ("Megascience – NICA"), No.18-02-40101.

The application of this software to automate the processes of task running were supported by the RFBR grant ("Megascience – NICA") No.18-02-40102.

References

[1] NICA – Nuclotron-based Ion Collider fAcility. URL: https://nica.jinr.ru (Date of reference: 30.08.2021).

[2] Tsaregorodtsev A. and the DIRAC Project. DIRAC Distributed Computing Services // J. Phys.: Conf. Ser. 2014. Vol. 513, No. 3. DOI: 10.1088/1742-6596/513/3/032096.

[3] Korenkov V., Pelevanyuk I., Tsaregorodtsev A. Integration of the JINR Hybrid Computing Resources with the DIRAC Interware for Data Intensive Applications // Data Analytics and Management in Data Intensive Domains. 2020. P. 31-46. DOI: 10.1007/978-3-030-51913-1_3

[4] Kutovskiy N., Mitsyn V., Moshkin A. et al. Integration of Distributed Heterogeneous Computing Resources for the MPD Experiment with DIRAC Interware // PEPAN. 2021. Vol. 52, No. 4.