ISSN 1995-0802, Lobachevskii Journal of Mathematics, 2018, Vol. 39, No. 9, pp. 1251–1261. © Pleiades Publishing, Ltd., 2018.

Cloud Service for HPC Management: Ideas and Appliance

D. V. Puzyrkov[1]*, V. O. Podryga[1,2], and S. V. Polyakov[1,3,4]*****

(Submitted by V. V. Voevodin)

[1]*Keldysh Institute of Applied Mathematics, Russian Academy of Sciences, Miusskaya pl. 4, Moscow, 125047 Russia*

[2]*National Research Center "Kurchatov Institute," pl. Akademika Kurchatova 1, Moscow, 123182 Russia*

[3]*National Research Nuclear University MEPhI, Kashirskoe sh. 31, Moscow, 115409 Russia*

[4]*Moscow Institute of Physics and Technology (State University), Institutskii per. 9, Dolgoprudnyi, Moscow oblast, 141701 Russia*

Received April 23, 2018

Abstract—The paper presents a cloud service aimed to solve promising nanotechnology problems on supercomputer systems. At the present stage of computer technology evolution it is possible to study the properties and processes in complex systems at molecular and even atomic levels. Such calculations require the use of high performance computing systems of various types, for example, GRID systems and HPC clusters. Considering the time consuming computational tasks, the need arises of software for automatic and unified monitoring of such computations, as well as unified environment for data preparation and storage. The preliminary result of the work is a prototype of the complex cloud environment, implemented as a KIAM Multilogin service and an application software accessible from users virtual machines. It consists of a virtual desktop environment, and the web-based application for calculation management, that includes project-management features, remote task execution abilities, storage and monitoring features and basic pre- and post-processing capabilities. The first applications of the service were the software packages GIMM_NANO and Flow_and_Particles, designed to solve the actual problems of nanoelectronics, laser nanotechnology, multiscale problems of applied gas dynamics. In this paper the main ideas, the cloud service based on, the inner structure of the service and the technologies used to build it are considered.

DOI: 10.1134/S1995080218090172

Keywords and phrases: *cloud service, virtualization, supercomputer modeling in nanotechnology problems, visualization, molecular dynamics.*

1. INTRODUCTION

The present work is devoted to the development of applied cloud services intended for solving computing management and data management tasks in large-scale simulations. Fundamental nature and relevance of the common problem comprise the fact that at present in context of introducing the nanotechnology in many industries there is an urgent need to combine various mathematical approaches information and computing resources into a unified tool of computer and supercomputer modeling. Because of a large amount of scientific software packages (for example for molecular dynamic simulations: LAMMPS, GROMACS, etc.) and a large pool of different HPC resources, available for a researcher, it became quite a problem to perform a proper management of large-scale multi-task computations, ones that use different software and hardware in a single scientific project. Furthermore, the results of each calculation or simulation, performed within a single project, have to be proper monitored, analyzed and catalogued.

*E-mail: dpuzyrkov@gmail.com

**E-mail: pvictoria@list.ru

***E-mail: polyakov@imamod.ru

The job scheduling environments such as SLURM helps to run and monitor such tasks, but only in case of a particular HPC resource, and not in case of multiple access points. And if a researcher run multiple computations over different clusters the task of monitoring consumes a lot of researchers time. The most successful way to solve a problem of unification of computing management, job scheduling and applications packages is to create relevant cloud environments and services in which each user can have access to all possible information materials, modeling programs, computing resources and industrial CAD. The specific task is to create a cloud service using heterogeneous clusters and supercomputers for industrial nanotechnologies.

2. MOTIVATION AND PROBLEMS

In this paper, the task was to create a cloud-based service for computing management and unified environment to use in promising tasks of nanotechnology.

The motivation for this task was the general trends in the development of software for mass use. Within the framework of cloud technologies, many problems are solved, such as hierarchical user authentication, the creation of a common protected information space, the work of users with personal desktops of virtual machines, migration of virtual machines in the constantly expanding and modifying field of calculators, optimal management of computing resources, convenience of developing specific applications, guaranteed secure storage of critical information, joint software development, training in work with services, and much more.

First of all, we should mention a wide field of applications that had to be integrated in such service (for example, ANSYS CFX, StarCD, OpenFOAM, FlowVision, AB INITIO, GAMESS, GAUSSIAN, VASP, HyperChem, LAMMPS, GROMACS and many others). It is important to note that not all of them can work in batch mode, and almost all of them have a GUI (Graphical User Interface) based on desktop technologies, that can not be used in web-application, and this fact automatically imposes a need for a full-fledged working environment, with common-use OS and ability to launch such applications.

Second, all of this software operates their own format of input and output data, and the task of its storage and cataloguing can not be solved in any different way than a simple file storage facility. Using different software in a single project also leads to a inconsistent of data storage and computational results. Such problem can be solved using project-management instruments, assigning each calculation or simulation to a special-formed request (a ticket), with all links and metadata described in it.

Third, the automation of calculation, involving multiple software packages in different combination on an HPC system requires a scenario system, that will be scalable and robust, and also will be easy to use by a common researcher. Also, usually there is a set of different supercomputer systems, available for a researcher. This systems may use different job-schedule systems, or can be unmanned at all, using simple MPI interface for calculations, like simple institutional clusters. To utilize them all, there is a need in software framework which is able to launch applications over selected hardware in simple, automatic way. For a good usability of this framework it is necessary to have an ability to relocate computational data from one HPC resource to another, in case of quota expiration, and automatic data retrieving for analysis in case of running calculation.

3. EXISTING SOLUTIONS

The closest to the proposed service concept is a concept of web-laboratories. Web-laboratories are cloud services, usually build on PaaS (plaform as a service) and IaaS (infrastructure as a service) principles. Such services usually provide a set of simulation tools or simulation frameworks in some way, and a HPC infrastructure to run such simulations on. They often provide collaboration abilities and some basic project-management features.

As a system for HPC access and monitoring we can mention the “FlyElephant” platform [1]. FlyElephant provides access to different HPC clusters and helps users to send their computing tasks to them. For now, the platform support has access to 3 clusters and it can work with SLURM. This platform is not aimed on particular scientific problems, providing users a set of common applications, libraries and frameworks for data science. Any special tasks, such as data generation for simulators should be done by other software.

As an example of web-laboratory service we can mention NanoHUB [2] web-laboratory aimed on nanotechnology simulations. Its frontend is based on java-applets, and its backend provides a sophisticated system that runs simulations over a grid-type HPC resources. It provides a user a set of web-interfaces for multiple simulation tools, that allows him to modify, run, and post-process them just using a web-browser.

Another similar solution is a UniHub platform [3]. It is a "web-laboratory" service based on IaaS and PaaS concepts. UniHub is built over OpenStack solution, and provides user a VDI interface for a personal desktop, and some web resources for collaborative work and education such as web-conference engine and report-creation environment. In case of such services usually there is no possibility for a researcher to use custom HPC resource, only those that are available in a system, as well as custom software for a simulation and engineering.

4. IDEAS

Considering the described above problems, the main idea of developed service is to provide a researcher a single access point to multiple scientific applications, HPC resources and give him abilities for project-management, automatic data relocation and computational monitoring features.

Basic functionality, such as pre-configured environment with multiple installed applications can be achieved using a VDI (virtual desktop infrastructure) approach. Since VDI operates with full-fledged desktop operation systems and allows to integrate in it almost every software package, it is possible to create a working environment customized for particular scientific problem or particular user. This infrastructure should be the mentioned above unified access point to the overall services, available in the project.

As for other parts of the developed service, such as project-management and computing-management, they can be integrated with each other in a cloud service, with a web-application for control and a server-side executors for all background work. This service will operate using four different frameworks, integrated with each other and providing a unified REST API for a web-application and different control applications.

First of all is a user management and project management system. Users in the developed service are defined by a VDI service, but should have a personal account in a management console. In case of different qualification of users it is necessary to limit their abilities by roles in the system. Each user should have his personal storage for data, and have an opportunity to register his credentials to an available HPC system. Project management service should be a web-based application, with links to all necessary resources such as documentation. Structure of the requests in it should reflect a scientific purpose of the service. A privileged user should be able to create a project, invite other people for collaborative work, and to allow the particular project to use selected HPC resources on behalf of him.

Second important part is a storage system and VCS (version control system). Taking into account that a lot of scientific problems require a special development of computational programs, their versioning is quite useful feature. It will allow to monitor the development process by a project supervisor as well as bug-tracking and easing of collaborative development. Also, there is a need in storage facilities, somehow attachable to each other, for all computational data, which the researcher will use. It can be either a personal storage space and an internal storage for ongoing computational data, received by monitoring system.

Third framework is a scenario system. This part of the cloud service should implement in some sort of script language an ability for user to describe a desired computational workflow. Requirements are simple: common script language, scalability and ease of use. The script should allow user to define inputs of a calculation, targeted output data, to link application output with another application input, to support conditions and cycles. In view that there is no possibility to upload all data and software to a target HPC system during preparation stage, this script should be performed on a backend-server and give commands for task runner system.

The last necessary framework is a task runner system with abilities for monitoring. Project management should be integrated with this system, which feature is a remote task execution on different HPC resources. It should also provide a monitoring capabilities in automatic way: data and task relocation from one HPC system to another, quota monitoring and on-fly post-processing of received dataset.

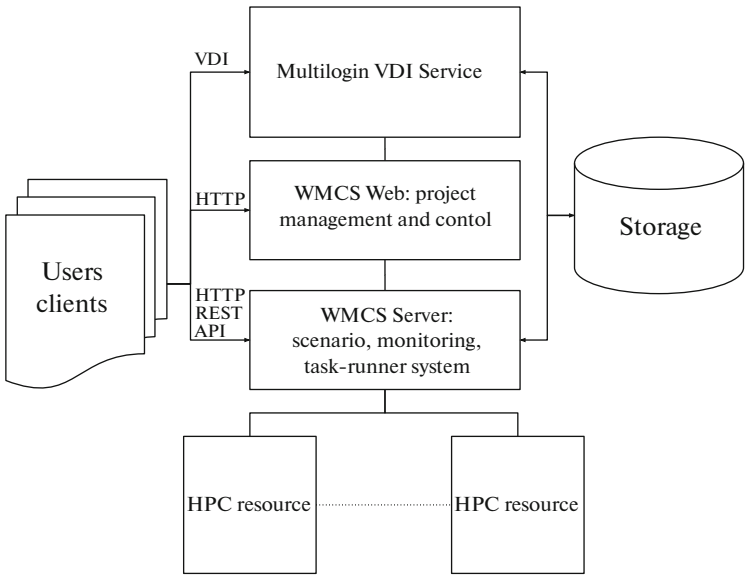

Fig. 1. Multilogin service basic architecture.

5. GENERAL ARCHITECTURE

The general structure of developed service is shown in Fig. 1.

The main access point to a system is a Multilogin VDI server. It gives the researcher an ability to use pre-installed software packages and maintain his personal storage in well-known desktop environments using a widespread operating systems, such as Linux and Windows.

Separately from VDI service there is special framework written in Python language, and a set of server applications, called WMCS, intended to solve a problem of data management and computing management. This software system is designed as a basic web service, with a user interface (UI) web-application and a backend services accessible through HTTP API interface.

Web-application server provides project-management abilities, management console and UI for scenario system. The backend server part is a service, which maintain a task-runner system, providing a REST HTTP API for a resource access, task running and data-management capabilities. Both of them can be placed on separate physical servers, for scalability reasons.

WMCS-Web is a common web-application, served by web-application server, provides user an ability for maintaining it's own projects and calculation. It was written in JavaScript language with backend part written in Python using Django framework.

Second part is a complex solution, written on Python (the WMCS framework) and based on Django web framework for REST API functionality. This part of a service implement an operational server provides all heavy-duty operations: resource access, calculations launch and monitoring, data-management, and scenario execution.

Such division of services allows to develop a command line interface for usage on personal workstation, if the project-management solution is unnecessary for a researcher.

As a VCS system we choose a git VCS, with an interface for it based on GitLab Community Edition software. Though GitLAB has a convenient API, it was integrated in the project-management software as well as in task runner system, providing a version-controlled storage for multiple datasets, software and calculations results. An idea of integrating a VCS in this service bases on a need to force a developer/researcher to store his application scripts and calculation scenarios in version control system. This is a good practice, because of that the researcher usually has to write different type of codes, such as configuration for simulation packages. VCS allows to eliminate a problem of sudden data loss, by accident of other user-related issues, and provides an opportunity for re-use of such codes in simple way.

5.1. Multilogin VDI Cloud Platform

KIAM Multilogin [4–6] cloud platform has been developed and implemented in the previous two years, it allows accessing to virtual desktop environment with pre-installed software packaged intended to solve some particular task, such as geometry modeling.

It assumes the existence of a distributed computing platform hidden from users and a single scheme for accessing it via the Internet/Intranet using a specialized web portal. Standard user authentication mechanisms (login-password, login-public key, etc.) are implemented here. This part of user authentication is a VDI-service only. Any user, registered in a system, can access to virtual desktop. The role-distribution rests on shoulders of WMCS framework.

Inside this cloud architecture, in addition to access servers, there are an information portal, a training portal, a research portal, a developer portal, and computing resources in the form of database servers and working computers (clusters and supercomputers).

User virtual machines (VM) are stored in one or more databases. They can be started by the service at any time and on any access server, but only in a single copy. Simultaneous launch of a user's virtual machine on multiple access servers is excluded. In this case, it is possible to migrate the user's virtual machine to a less-loaded access server, performed by storing all VM data and cold restart. Also, each user can have several variants of VM, distinguished by the type of OS and/or and intended purpose as well as several saved states of a particular VM (snapshots). After passing the initial training course, the user gets the status of a user-researcher and/or a developer user and an administrator grants him an access token to the WMCS API service. At this stage all WMCS resources and services can be accessed directly.

5.2. WMCS Framework

This part of developed service provides a registered and trained user an environment for maintaining his research or engineering tasks. It is accessible from the Internet/Intranet like a VDI service, but user authorization is performed by security token, gained by him from system administrator after train course. Using this service, a researcher is capable to create his personal projects, register his own HPC systems and create calculation scenarios, using available software. This part of the service implemented as two common web-applications integrated with each other. Both of them are written in Python language within Django web framework.

First application is a web-application in general sense: it provides frontend, and gives html+JavaScript page for client's browser. It implements an access portal for internal resources (projects, applications, datasets) database and provides a proxy access to another part of WMCS service, responsible for heavy operations such as task-monitoring. The purpose of such division (of course, it could be developed as a single web-service) is to give an ability to connect multiple access-servers (for example that belongs to different organizations) to a single operational server. Any organization can install this software on their own hardware, protect it by their security system (or put it in intranet) and connect the system to somebody else operational server.

The second application is an operation server part. It is a service that responsible for all task-management jobs. It consists of a web-service providing public REST HTTP API, a scenario system implementation, integrated with VCS and project management. Running a calculation makes this service to create a monitoring process, that uses a scenario, defined by user, to send commands on an HPC system.

6. PROJECT MANAGEMENT

Project management is a basic management approach for a big software system aimed for multiuser collaboration work. It is the practise of planning, executing and controlling the work of a group of collaborative users to achieve specific goals. Main ideas of this technique are to divide users by their role in a project and to give a supervisor an ability to monitor an control the overall progress.

This approach forces a supervisor to describe tasks and goals for the research computations in understandable way, and helps to proper manage the results and datasets because of their belongings to a described projects. Every task, user can perform on HPC system, is described by a calculation entity, assigned to a specified project entity. A project entity aimed to enclose all calculations performed to archive an intended goal. It encapsulates tasks, documentation, collaborators and results for a specific problem.

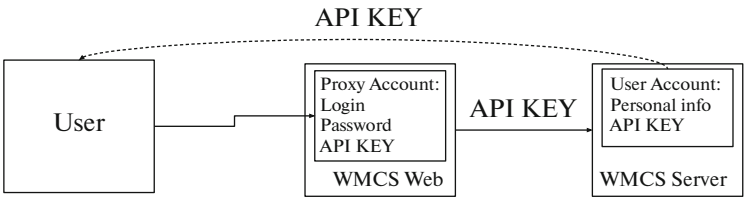

Fig. 2. User access scheme.

6.1. User Management

First part of any multiuser system is user management, authentication and authorization. As mentioned above, WMCS-Web provides a GUI for management server and performs proxy calls to operational server, so in this application no user information is stored, except a so-called "proxy account", containing an API authentication token for operational server. Once a user gives the system his API key it creates a proxy account for this user, checks the provided credential and updates internal information for limiting GUI capabilities suitable for a user role. Simplified user access to a WMCS-service scheme is shown in Fig. 2.

All user control is performed by WMCS operational server. Concept for user management based on common web-application model: user account, user role, personal key. First of all, every user of the WMCS has his own account. It contains basic personal information.

Users are created in database by a system administrator, using a ticket-system. After the account was created, a user is informed by e-mail or any other ways, and provided with his authentication token, that allows him to use a all available functions.

Users are divided in different roles: Administrator, User and Developer. This roles gives him an access to different parts of the system. For example, Developer has an access to a VCS system, that makes him possible to update and develop all software used in WMCS, such as scientific applications, visualization scripts, etc, Administrator can commit the changes, made by developer to a production code-base. User is a simple account that can create his own projects and use available software, without an ability to change internals of a system.

6.2. Project Entity

Project is a accumulative entity, based on a concept of targeted research. If a scientist has to do any calculation using WMCS system, he should create a project page, fill the description of it and link all related data: documentation, etc. After the research he also can link results of it to a project, and get a report: calculations, status, result data, parsed results. This concept helps to manage multiple different calculations aimed to solve one specific task. Multiple users can work over single project. A collaboration capabilities over a single research include a ticket-based system to solve management problems. The results of a project can be as linked data from particular calculation, or a report created by user. Project relations and entities are shown in Fig. 3.

6.3. Calculation Entity

Calculation entity describes the particular task that should be performed on HPC system. It consists of a description, a scenario in Python language, describing a workflow for this computation and a set of input parameters. It also contains a set of documentation files, for applications and tasks. Part of them are inherited form the project, but it also can contain custom documents describing the process will be performed. Each calculation entity can use an output (a result) of another calculation in a project this entity belong. For a calculation it is possible to chose a HPC resource, from available for a project, on which it will be performed.

It is also possible to create a VCS repository for a calculation data and metadata, to enable version control for results and sources code of applications in use. Any post-processing tasks operate over calculation entity. The inner part of this objects are scenarios implemented in Python language, that describe the applications they will use and their interaction in terms of files an command line parameters. Monitoring system is also implemented in a way to operate with this objects. Only calculation can be ran, monitored or processed in any system-related application.

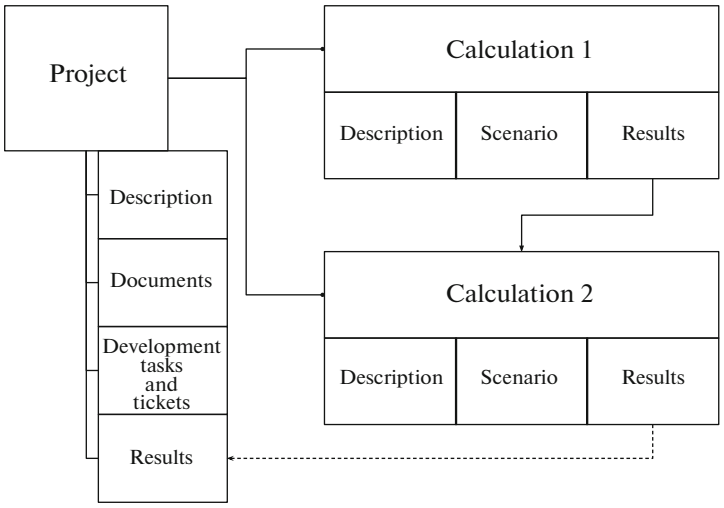

Fig. 3. Project and calculations relation scheme.

7. COMPUTING MANAGEMENT

Except a user database and project management feature, main task of a server part of WMCS is a computing management. This part of its functions is mostly hidden from user, and most of the settings are accessible only for system administrator.

First of all it is a scenario and application-use framework, that allows user to write a calculation scenario, based on registered in system applications and input datasets. It uses Python as script language. This part is a starting stage of WMCS framework workflow, defining which applications should be used in calculation, what sequence of applications launches should be performed and what data should be set as input for each application.

Second, WMSC task-runner subsystem called KIAM Job_Control [5] is responsible for remote task execution and monitoring. This framework is capable to launch selected applications on remote HPC resources, such as supercomputers and HPC clusters. Scenario-system uses this framework to launch specified application on target resource. It defines an application and dependencies, and the Job_Control system does the data transferring stuff and on-resource commands execution.

Third part of computing management is a calculation process control and monitoring. In case of special applications, developed using WMCS-monitoring-library (available in C, C++, Fortran and Python languages) this process is native: WMCS gives a working process a command using special interface (mostly files, but there can be other ways).

In case of generic application, the WMCS can only use an OS process management features: run tasks, kill tasks and monitor their statuses by PID. It also supports job-queue environments like SLURM and SUPPZ.

Last task of computing management system is a data relocation procedures: moving computations from one resource to another, and on-fly downloading and post-processing of ongoing computation. Applications, that were developed with a special WMCS-monitoring-library define by themselves what data is necessary to relocate or can be post-processed. For general applications such functionality is provided by describing necessary parameters in so-called “application passport” metadata.

7.1. Applications

Any application, a researcher can use, should be registered in a WMCS system. A registration procedure is available in two modes: system-wide, available for a developers and approved by system administrator, and user-space, available for a particular user or project. For a particular software, a researcher wants to use, the registration workflow is following. User creates a VCS repository, where he stores all necessary data for this application: binary files, scripts, documentation, and describes the software usage procedures in “application passport” file. “Application passport” is a metadescription of a particular software. An example of such file is shown in listing 4. Using this file researcher describes

```
run: bash run.sh
build: bash make.sh
cwd: ./RUN/
input:
    configuration files:
        type: static
        format: dir
        path: ./RUN/configs/
        description: default customization files

    solver config:
        type: static
        format: file
        path: ./RUN/usertask
        description: main solver configuration file
output:
    outdir:
        type: dir
        path: ./RUN/OUT/
    logdir:
        type: dir
        path: ./RUN/LOG/
    vizdir:
        type: dir
        path: ./RUN/VIZ/
    mesh:
        type: dir
        path: ./RUN/MESH/
    control file:
        type: file
        path: ./RUN/control.dat
```

Fig. 4. Passport.

such parameters as run command, build command, input parameters and output datasets, produced by this software. For basic packages, included in Multilogin service, such passports were already created.

This "passport" is intended for scenario system, as well as for task-runner system. It is written in YAML language to provide a user-readable structure. Scenario, that uses this application, defines the links between this application and possible other applications. Input parameters for any applications can be certain type and format. It can be declared as file, directory or command-line argument. Output datasets presumed to be files or directories. At present stage of development of this service it is necessary for applications to be compatible some how with each other by file interface.

A WMCS-Web application (and WMCS Server) provides user an ability to modify input files and parameter using special API. When user writes a scenario using an application, WMCS parses this scenario, extracts applications and creates a virtual file space for them. Every input/output data, described in passport is available for modifications. Any other application data is hidden from user.

7.2. Scenarios

Scenario script is a description of computational process in Python language. It operates with applications, registered in WMCS system. Writing this script, user defines a software involved in a calculation, their interactions and workflow. Scenario is performed over the operational server hardware, and through it execution commands a task-runner and monitoring systems what software and data should be deployed, build and run on a HPC resource. In case of multi-application scenario it is possible to assign different HPC resources to different calculation branches. If there are multiple HPC resources are available for a calculation, the system tries to distribute parallel branches over different available resources. A sample scenario for multi-application calculation is defined in listing [5].

It operates with five different applications (but they should be synchronized by input-output format): a mesh generator (RectangularMeshGenerator), an application that sets initial conditions (StartConditionSetter) on generated mesh, two different solvers (MySolver1 and MySolver2) and a post-processor

```
import wmcs
meshgen = wmcs.get_application ("RectangularMeshGenerator")
initcond = wmcs.get_application ("StartConditionSetter")
solver1 = wmcs.get_application ("MySolver1")
solver2 = wmcs.get_application ("MySolver2")
postproc = wmcs.get_application ("MyVisualizer")


def applications ( ) :
    return [meshgen, initcond, solver1, solver2, postproc]

def postprocess ( app):
    calc = postproc.new ( )
    calc.set_inputs (app.get_outputs ( ))
    calc.run( ).wait ( )

def outputs ( ) :
    return [postproc.get_outputs ( )]

def run( ) :
    mgcalc = meshgen.new ( )
    mgcalc.run( ). wait( )

    initcalc = initcond.new( )
    initcalc.set_inputs (mgcalc.get_outputs ( ))
    initcalc.run ( ). wait ( );
    solvers = [solver1.new ( ), solver2.new ( )]

    for solver in solvers :
        solver.set_inputs (initcalc.get_outputs ( ))
        solver.run (on_data_signal = postprocess)

    for solver in solvers:
        solver.wait ( )
```

Fig. 5. Scenario file for WMCS.

application (MyVisualizer) that can visualize output datasets of solvers. Every application is registered in WMCS through VCS system, and has their own input and output sets of parameters, defined in their passports. These inputs and outputs can be automatically assigned to another application. If no additional information is provided, the system compares input-output datasets by name and type, and links them if they are consistent. WMCS provides an ability for a "dry-run" of a scenario, to generate a graph of input-output dependencies. If there is some indeterminacy in that graph, system will generate an error and suggestions how a researcher can fix them.

In the example, the output of a mesh generator is linked to the inputs of conditions setter, and the output of this application is linked to both of solvers. The execution branches are parallel for solvers and post-processor, so this part of calculation could be performed over different HPC resources. As the solvers are implemented using the WMCS-monitoring library, it is possible to launch a post-processor application every time they generate certain signal.

All unlinked input parameters of this calculation (such as solver configuration) are accessible and modifiable through WMCS Server API and in WMCS-Web UI.

7.3. Appliance

Figure 6 shows an prototype of a GUI for the service. In this particular case it operates over one test-purpose project with one calculation associated for it.

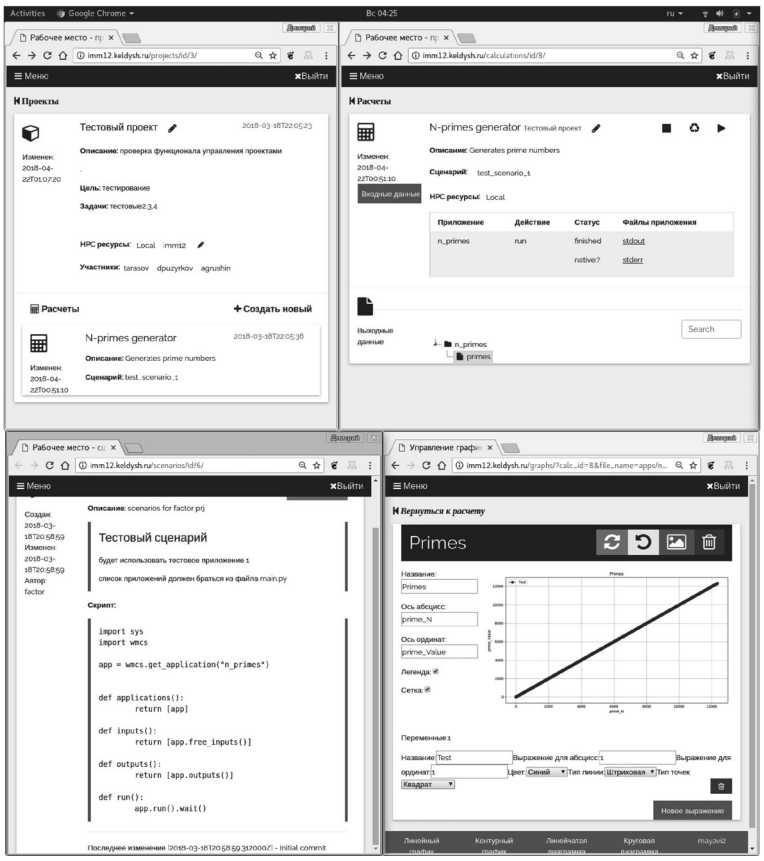

Fig. 6. A GUI prototype for WMCS-web: projects and calculation page, particular calculation page, a scenery brief, and a 2d-plots for data analysis.

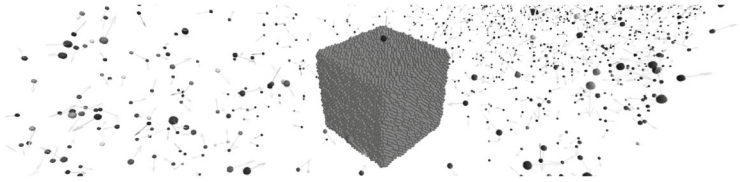

Fig. 7. Visualization of a gas-metal microsystem, calculated using the Multilogin service.

User creates a project, choose from available scenarios and create a calculation entity. When a calculation is performed (or WMCS got a signal, that new data is available) a new entry appears in "output data" section, which can be processed and analyzed in another calculation, or a visualizer.

The GUI is still in development, but back-end part already can perform this management in command-line mode. It was tested using a GIMM_NANO [7] package for MD simulations [8, 9]. It allows us to describe the whole computational model and run different calculation over a single project: to create a microsystem that contains a metal cube and a gas, floating around it [10, 11]. The parts of this project were such as initial dataset creation, calculation of thermodynamic equilibrium of both materials, constructing a new geometry using the computed states and running a simulation of a gas flow. Using the developed service we achieved a simple on-fly monitoring of both calculations intended to calculate a thermodynamic equilibrium performed over different HPC resources. It also allowed us to find an error in a simulation code thanks to on-fly post-processing capabilities. The visualization of a final calculation is shown in Fig. 7.

8. CONCLUSION

The paper presents a cloud service, intended for solving promising nanotechnology problems on supercomputer systems. The prototype is implemented as an access service KIAM Multilogin and WMCS software package for maintain calculation and project management. The first experience of parallel applications integration within the framework of the created service was using separate programs from the GIMM_NANO and Flow_and_Particles packages intended for solving complex problems of nanoelectronics, laser nanotechnology, and multiscale problems of technical gas dynamics. Further development of the service consists in improving the management systems of information, educational and computing resources, adding new packages and applications, creating a flexible project management system, increasing the capabilities of the visualization system.

ACKNOWLEDGMENTS

The work was supported partially by Russian Foundation for Basic Research, projects No. 15-29-07090-ofi_m, 16-07-00206-a, 17-01-00973-a. This work was partially supported by the Academic Excellence Project of the NRNU MEPhI under contract with the Ministry of Education and Science of the Russian Federation No. 02.A03.21.0005.

REFERENCES

1. FlyElephant Website `https://flyelephant.net/`.
2. G. Klimeck, M. McLennan, S. P. Brophy, G. B. Adams III, and M. S. Lundstrom, "nanoHUB.org: advancing education and research in nanotechnology," Comput. Sci. Eng. **10** (5), 17–23 (2008).
3. O. I. Samovarov and S. S. Gaysaryan, "The Web-laboratory architecture based on the cloud and the UniHUB implementation as an extension of the OpenStack platform," Tr. ISP RAN **26**, 403–420 (2014).
4. S. Polyakov, A. Vyrodov, D. Puzyrkov, and M. Yakobovskiy, "Cloud service for decision of multiscale nanotechnology problems on supercomputer systems," Tr. ISP RAN **27**, 409–420 (2015).
5. D. Puzyrkov, V. Podryga, S. Polyakov, and M. Iakobovskii, "KIAM JOB_CONTROL task management environment and its application to cloud and GRID computing," CEUR Workshop Proc. **1787**, 416–422 (2017).
6. M. V. Yakobovskiy, A. A. Bondarenko, A. V. Vyrodov, S. K. Grigoriev, M. A. Kornilina, A. I. Plotnikov, S. V. Polyakov, I. V. Popov, D. V. Puzyrkov, and S. A. Sukov, "Cloud service for solution of multiscale nanotechnology problems on clusters and supercomputers," Izv. IuFU, Tekh. Nauki **12**, 103–114 (2016).
7. A. A. Bondarenko, S. V. Polyakov, M. V. Yakobovskiy, O. A. Kosolapov, and A. M. Kononov, "Software package GIMM_NANO," in *Proceedings of International Supercomputer Conference Scientific Service on the Internet: All Facets of Parallelism, Sept. 23–28, 2013, Novorossiysk, Russia*, pp. 333–337.
8. V. O. Podryga, "Calculation of kinetic coefficients for real gases on example of nitrogen," Lect. Notes Comput. Sci. **10187**, 542–549 (2017).
9. V. O. Podryga and S V. Polyakov, "Molecular dynamics simulation of thermodynamic equilibrium establishment in nickel," Math. Models Comput. Simul. **7**, 456–466 (2015).
10. D. Puzyrkov, V. Podryga, and S. Polyakov, "Parallel processing and visualization for results of molecular simulations problems," Proc. ISP RAS **28**, 221–242 (2016).
11. V. Podryga and S. Polyakov, "Calculation of nitrogen flow in nickel micronozzle based on numerical approaches of gas and molecular dynamics," in *Proceedings of the V International Conference on Particle-Based Methods. Fundamentals and Applications, Particles 2017, Sept. 26–28, 2017, Hannover, Germany,* pp. 744–754.