ИПМ им.М.В.Келдыша РАН • Электронная библиотека

Препринты ИПМ • Препринт № 50 за 2022 г.

ISSN 2071-2898 (Print)
ISSN 2071-2901 (Online)

Н.И. Тарасов

# Архитектура и реализация цифровой платформы для проведения вычислительных экспериментов на суперкомпьютерах



***Рекомендуемая форма библиографической ссылки:*** Тарасов Н.И. Архитектура и реализация цифровой платформы для проведения вычислительных экспериментов на суперкомпьютерах // Препринты ИПМ им. М.В.Келдыша. 2022. № 50. 30 с. https://doi.org/10.20948/prepr-2022-50
https://library.keldysh.ru/preprint.asp?id=2022-50

**Ордена Ленина**
**ИНСТИТУТ ПРИКЛАДНОЙ МАТЕМАТИКИ**
**имени М.В.Келдыша**
**Российской академии наук**

**Н.И. Тарасов**

# Архитектура и реализация цифровой платформы для проведения вычислительных экспериментов на суперкомпьютерах

**Москва — 2022**

*Тарасов Н.И.*

**Архитектура и реализация цифровой платформы для проведения вычислительных экспериментов на суперкомпьютерах**

Рассмотрены современные подходы, применяемые при конструировании высокопроизводительных аппаратно-программных платформ коллективного пользования. Предложена архитектура веб-системы, позволяющая существенно упростить использование прикладного программного обеспечения, предназначенного для математического компьютерного моделирования физических процессов сложной природы на суперкомпьютерных устройствах. Выработан технологический стек, на основе которого разработан продемонстрированный прототип цифровой платформы. С помощью разработанной платформы решен ряд задач, связанных с очисткой водной и воздушной сред от вредных примесей, содержащих соли металлов.

***Ключевые слова:*** веб-интерфейс, цифровая платформа, супер-компьютерные вычисления

*Nikita Igorevich Tarasov*

**Architecture and implementation of a digital platform for numerical experiments on supercomputers**

Modern approaches used in the design of high-performance hardware and software platforms for collective usage are considered. A web-system architecture has been proposed. Its main goal is to significantly simplify the use of software designed for mathematical computer modeling of physical processes of a complex nature on supercomputer devices. The technological stack has been developed, on the basis of which the digital platform prototype of has been developed. With the help of the developed platform, a number of tasks related to the purification of water and air from harmful impurities containing metal salts have been solved.

***Key words:*** web-interface, digital platform, supercomputer calculation

Работа выполнена при поддержке грантов РФФИ 19-31-90140_Аспиранты и 20-51-18004_Болг_а.

# Введение

С ростом производительности центральных процессоров, а также применением видеокарт в качестве базовых вычислительных устройств стало возможным решение многих сложных научных и инженерных задач на персональных компьютерах и небольших системах коллективного пользования. Одновременно с этим шло развитие статистических и прямых методов математического моделирования, которые смыкаются в настоящее время с технологиями нейросетей и искусственного интеллекта. В результате объединение традиционных и новых подходов в математическом и компьютерном моделировании стало повседневной практикой. Обычная и высокопроизводительная техника становится все более доступной во многих исследовательских организациях и на производстве. В связи с этим возникает задача создания аппаратно-программных платформ для решения большого круга научно-технических проблем. Основной целью подобных систем является существенное упрощение и автоматизация большинства этапов проведения вычислительного эксперимента, включающих подготовку исходных данных, управление и слежение за расчетным процессом, анализ результатов расчета. Реализация современных цифровых платформ, безусловно, должна учитывать тенденции развития вычислительной техники, в том числе высокопроизводительной.

Основу вычислительных кластеров и суперкомпьютеров сегодня составляют гибридные системы, построенные в рамках модели распределенных вычислений, в которой множество гибридных вычислительных узлов (имеющих в своем составе центральные процессоры и альтернативные устройства обработки данных, например графические ускорители) объединяется высокоскоростной сетью [1]. Примером таких систем служат гибридные вычислительные кластеры ИПМ им. М.В. Келдыша РАН – К100 и К60 [2]. Именно такие системы в конечном итоге способны развиваться продолжительное время и адаптироваться к новым все возрастающим потребностям в вычислениях, допуская увеличение вычислительного поля как посредством увеличения количества узлов, так и модернизации уже существующих.

В части системного программного обеспечения (ПО) современные гибридные кластеры и суперкомпьютеры управляются UNIX-подобными операционными системами, обеспечивающими многозадачный и многопользовательский режим работы, в том числе в рамках удаленного доступа, при соблюдении всех необходимых стандартов безопасности. При этом для эффективной организации параллельных вычислений используются различные интерфейсы и средства параллельного и гибридного программирования (MPI, OpenMP, CUDA Toolkit и др.), а также системы управления заданиями пользователей. Основная цель последних – планирование вычислений, обеспечение доступа к вычислительным узлам кластера,

организация очереди и управление расчетными заданиями пользователей (от планового выделения ресурсов и запуска заданий, до слежения за состоянием конкретного задания и гарантированного сохранения результатов проводимых в его рамках расчетов). Примером такой системы в России является СУППЗ [3], разработанная сотрудниками ИПМ им. М.В. Келдыша РАН и ФГУП «НИИ «Квант». Широко известны и распространены также альтернативные программы организации очереди, такие как Slurm [4].

Стоит отметить также современный подход, основанный на технологиях облачных вычислений, представляющий альтернативу классическим системам совместного использования вычислительных мощностей, основным принципом которого является предоставление ресурсов по требованию посредством слоя виртуализации. Пользователю предлагается либо доступ к виртуальной машине, контейнеру Docker [5], системе контейнеров Kubernetes [6] посредством стандартного SSH-соединения, либо специализированный сетевой программный интерфейс приложения (API). Сегодня на рынке существует множество решений, предоставляющих услуги облачных вычислений, таких как Microsoft Azure [7], Amazon Web Services [8], отечественным аналогом которых может выступать Yandex Cloud [9]. Каждый из указанных сервисов предлагает собственную систему вызовов для выделения ресурсов, запуска и слежения за вычислительным процессом.

Помимо системного ПО требуется поддержка в виде библиотек и прикладных параллельных программ, позволяющих решать задачи из разных областей вычислительной и прикладной математики. Немаловажную роль при этом играют средства и технологии доступа к вычислительным ресурсам, средства подготовки расчетных данных и средства постобработки и визуализации получаемых результатов. При отчуждаемости прикладного ПО необходимо также обеспечить конечного пользователя отзывчивым графическим интерфейсом.

## Архитектура цифровой платформы

Текущий уровень развития веб-технологий позволяет реализовать комплексную цифровую платформу для проведения расчетов, нацеленных на суперкомпьютерное моделирование сложных физических процессов [10, 11]. Немаловажным фактором стало то обстоятельство, что при создании полноценного графического интерфейса, ориентированного на пользовательский компьютер, существует проблема производительности последнего, а также качества предустановленного на нем программного обеспечения. Применение веб-технологий исключает данные вопросы – для использования приложения достаточно наличия современного веб-браузера. Вся тяжесть предобработки и графического представления данных ложится на сервер, в качестве которого может выступать суперкомпьютер, на базе которого были получены основные результаты моделирования.

Помимо сказанного существуют дополнительные преимущества клиент-

серверного подхода, которые включают:

– обновление программного обеспечения не зависит от конечного пользователя, так как данная проблема находится в области ответственности администратора сервера;

– возможность совместной работы пользователей над одним расчетным проектом (многопользовательский режим);

– независимость прикладной программной среды в целом от состояния персонального компьютера пользователя, что особенно актуально при мониторинге прохождения задач на удаленных вычислителях и своевременной обработке расчетных данных;

– унификация доступа к вычислительным ресурсам различного типа в рамках единого подхода.

Основные существующие реализации подобных веб-платформ можно разделить на две категории. С одной стороны, это веб-лаборатории, такие как [12, 13], позволяющие проводить суперкомпьютерное моделирование в рамках отзывчивого пользовательского веб-интерфейса. Однако подобные системы направлены на решение конкретного класса задач. Кроме того, они не предусматривают свободного динамического расширения интегрированных вычислительных устройств.

Другой тип систем [14, 15] предлагает удобное администрирование вычислительных ресурсов и динамическое добавление прикладных программных кодов. Однако в этом случае конфигурирование прикладных программ осуществляется посредством стандартных строковых и числовых полей, флагов. Кроме того, доступ к выходным данным прикладных приложений реализуется посредством стандартного файлового интерфейса, обеспечивающего простейший просмотр или загрузку для локальной обработки.

В настоящей работе при создании прототипа цифровой платформы были учтены все вышеуказанные обстоятельства и приняты решения, направленные на создание своего рода вычислительной среды, в которой будет возможно проведение полномасштабных компьютерных и суперкомпьютерных экспериментов по выбранной прикладной тематике на доступных гибридных вычислительных кластерах в режиме удаленного доступа. Для этого предпринята попытка объединения представленных подходов. Структурная схема веб-платформы изображена на рис. 1 и отражает основные требования, накладываемые на разрабатываемую систему:

– многопользовательский доступ по протоколу HTTP/HTTPS;

– взаимодействие с удаленными вычислителями по SSH с возможностью расширения поддерживаемых протоколов (например, специализированного API конкретных решений для облачных вычислений);

– авторизованный доступ пользователей к системе, разделение пользователей по ролям (уровням доступа);

– обеспечение клиентов графическим интерфейсом для

взаимодействия с удаленными вычислителями;

– обеспечение информационной поддержки о состоянии вычислителя, прикладных программных кодах, проведенных расчетах;

– наличие расширяемой базы данных.

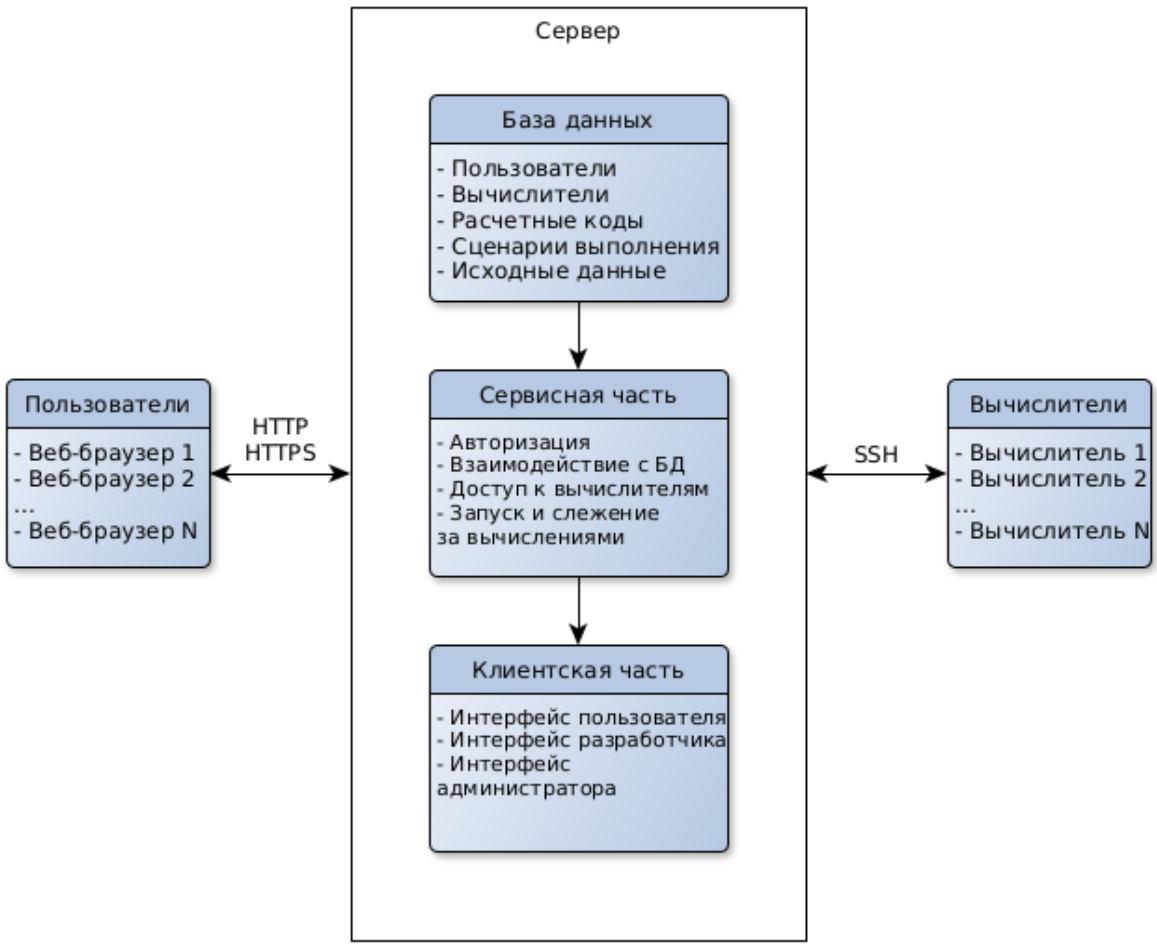


*Рис. 1.* Структурная схема цифровой платформы

Рассмотрим подробнее особенности и возможности системы, реализуемые выполнением основных требований.

В части безопасности, основывающейся на использовании шифрованного протокола HTTPS, веб-система позволяет исключить необходимость прямого доступа конечного пользователя к вычислительным ресурсам. Этот факт позволяет применить организацию сетевой инфраструктуры, в рамках которой суперкомпьютер или кластер располагается во внутренней сети, содержащей также веб-сервер платформы. При использовании подобного подхода для организации удаленного доступа достаточно разрешить использование во внешней сети порта, прослушиваемого веб-сервером. Поскольку цифровая платформа обрабатывает известный ограниченный набор запросов и сценариев

взаимодействия с вычислительным ресурсом, существенно сокращается количество возможных направлений атаки.

Разделение ответственности пользователей поддерживается посредством следующих основных ролей: обычный пользователь (расчетчик), разработчик программного обеспечения и администратор системы. В зависимости от типа аккаунта веб-платформа после авторизации предлагает модификацию графического веб-интерфейса, посредством которого реализуется соответствующий роли функционал. Таким образом, расчетчику предоставлены возможности проведения вычислительных экспериментов, разработчику – регистрация в системе новых прикладных приложений, администратору – обслуживание системы и вычислительных ресурсов, администрирование базы данных.

Классический процесс проведения комплексного вычислительного эксперимента заключается в ручной подготовке начальных данных локально или удаленно с помощью SSH-клиента в терминале. При локальной подготовке полученные файлы необходимо поместить на удаленный вычислительный ресурс. После этого посредством выполнения консольных команд производится запуск задания в соответствии с подходом, определенным на вычислителе (СУППЗ, slurm и т.д.), после завершения полученные данные необходимо проанализировать, зачастую локально.

Использование цифровой платформы расчетчиком для проведения вычислительного эксперимента предполагает подготовку исходных данных, запуск, управление и слежение за прохождением вычислительного задания и анализ полученных результатов с помощью графического пользовательского веб-интерфейса, без необходимости использования терминала и знания особенностей системы управления заданиями. Более того, набор приложений, выполняющих конкретную задачу (примером может служить процесс генерации сетки с последующим ее разбиением) в системе предлагается объединять посредством сценариев исполнения – специальных текстовых файлов, описывающих процедуру и порядок запуска. При этом все проведенные расчеты требуют наличия метаинформации, включающей название, текстовое описание, указание на используемый вычислительный ресурс, пути расположения файлов, сохраняемой в базе данных и доступных расчетчику как во время, так и после проведения расчета.

Разработчикам прикладных приложений применение предложенной клиент-серверной архитектуры позволяет проводить совместную разработку, ставшую практически стандартным подходом при разработке комплексных приложений. Дополнительным преимуществом является возможность обновления приложений для всех пользователей системы.

## Технологический стек

Первая разработка подобной веб-лаборатории [16 – 18] основывалась на Django [19] – библиотеке для языка программирования Python, содержащей все

необходимое для создания клиент-серверного решения, в том числе механизм объектно-реляционного отображения (ORM) для взаимодействия с базами данных, генератор веб-страниц из html-шаблонов, системы маршрутизации и авторизации. В процессе разработки выяснилось, что использования подхода шаблонных страниц недостаточно для создания интерактивного пользовательского интерфейса, обеспечивающего необходимые для проведения вычислительного эксперимента возможности. Для поддержки требуемого функционала приходилось использовать JavaScript код, исполняемый непосредственно в веб-браузере. В итоге от использования шаблонов было решено отказаться в пользу разработки одностраничного приложения (SPA) в реактивном стиле, для чего была применена библиотека Vue.js [20].

В данной разработке было решено исключить также и Django, поскольку большая часть его функционала, заключающаяся в шаблонах веб-страниц, перестала использоваться.

Программной основой для серверной части системы был использован Node.js [21], являющийся расширением JavaScript. Обычно язык программирования JavaScript используется на стороне веб-клиента и поддерживается во всех современных браузерах. Однако он имеет существенные ограничения по функциональности, в том числе невозможность взаимодействия с устройствами ввода-вывода и системными потоками нарямую. Для расширения JavaScript до языка общего назначения в целях создания серверного решения и используется Node.js.

Применение Node.js позволило, с одной стороны, добиться однородности кодовой базы между клиентской и серверной частью, а с другой – использовать высокую степень асинхронности JavaScript на сервере. Это особенно полезно для целей слежения за прохождением задач на удаленных вычислителях, выполнении длительных заданий и масштабирования системы.

Для обеспечения маршрутизации, а также использования плагинов безопасности был использован фреймворк Express.js [22], авторизация реализована посредством библиотеки passport.js [23]. Взаимодействие с базой данных осуществлялось с помощью библиотеки ORM Sequelize [24]. В качестве базы данных на стадии разработки использовалась SQLite [25].

Поскольку создаваемая цифровая платформа является закрытой для общего доступа, было решено организовать клиент-серверное взаимодействие на основе веб-сокетов [26]. Веб-сокеты представляют из себя надстройку над протоколом HTTP. В отличие от исходного протокола, они поддерживают соединение после начального клиентского запроса о подключении. При успешном установлении такового клиент и сервер могут произвольно обмениваться сообщениями. Сообщения, отправляемые посредством веб-сокета, включают в себя исключительно тело запроса, а HTTP-заголовок повторно не отправляется, что позволяет существенно оптимизировать клиент-серверное общение. Для организации веб-сокетов на сервере применялась библиотека ws [27]. Обеспечение HTTPS-протокола достигалось посредством обратного прокси

сервера nginx [28].

Клиентская часть была реализована с применением реактивной парадигмы, организованной с помощью использования фреймворка Vue.js и библиотеки компонентов Quasar [29]. Для SPA-маршрутизации применялось естественное в данном случае решение Vue-router [30], для организации централизированного хранилища данных использовалась Vuex [31] (хранение информации о пользователе, открытого веб-сокета, уведомлений). В отношении компонента анализа результатов расчетов использован ParaViewWeb [32] с встраиванием клиентской части в реализации Visualizer.js [33]. Данное решение обеспечивает поддержку форматов VTK, возможность интерактивной серверной или клиентской отрисовки, в зависимости от аппаратного обеспечения пользователя. ParaViewWeb поддерживает большое количество фильтров данных, сопоставимое со стандартным «толстым» приложением ParaView [34]. Отдельным преисуществом является возможность реализации собственного клиента на стандартном стеке цифровой платформы, отказавшись от встраивания Visualizer.js.

## Структура базы данных

База данных прототипа содержала минимально необходимый для организации предложенного механизма работы цифровой платформы набор таблиц, схема взаимосвязей между которыми представленна на рис. 2.

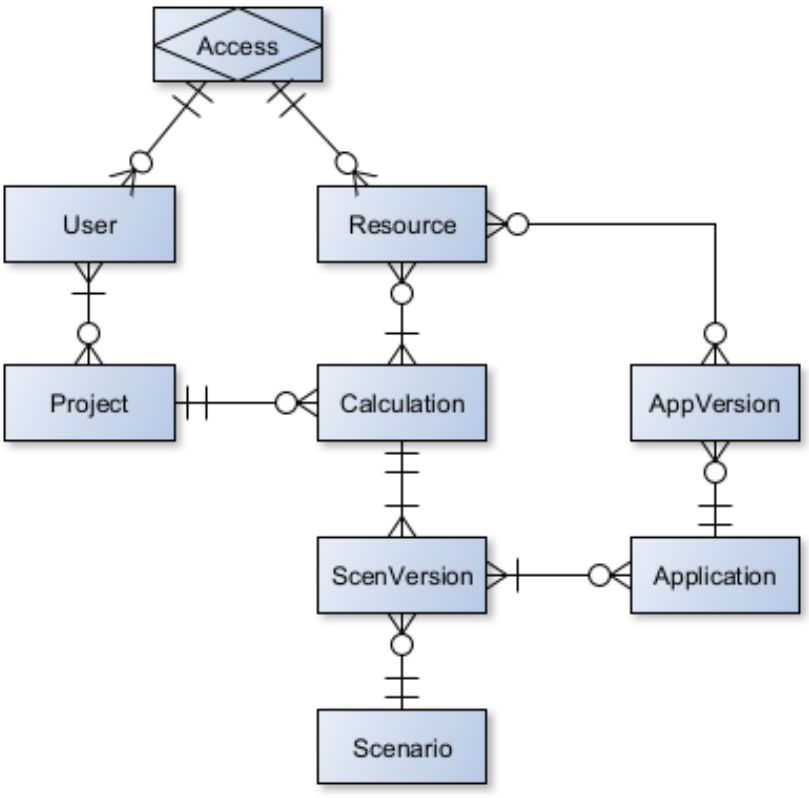


*Рис. 2.* Схема связей между сущностями базы данных

Для поддержки возможности динамической регистрации удаленного вычислительного ресурса необходима информация об адресе вычислительного ресурса, порте SSH-сервера, системе прохождения задач, возможностях задания переменных окружения. Эта информация размещается в базе данных цифровой платформы в таблице «Resource». Для осуществления SSH-доступа к удаленному вычислителю и разделения на нем пользователей используется таблица «Access», содержащая ссылки на записи пользователя системы и вычислителя, а также дополнительные переменные окружения и указание на используемые пользователем версии приложений. Подключение к удаленному вычислителю производится с использованием RCA-ключа веб-системы.

Описание приложений, кроме информационного, включает в себя также задание шаблонного метафайла, содержащего инструкции сборки, запуска, описание входных и выходных параметров, хранение которого производится в таблице «Application». Поддержка версионирования достигается с помощью таблицы «AppVersion», содержащей дополнительно информацию о пути к исходному коду и итоговый метафайл, используемый для генерации графического пользовательского интерфейса.

Сценарии (таблица «Scenario») используются для объединения нескольких приложений в единый цикл запуска с возможностью связывания входных и выходных данных приложений, для описания используется файл определенной структуры. Версионирование происходит при помощи таблицы «ScenVersion», итоговое метаописание которого используется при генерации графического пользовательского интерфейса настройки прикладного ПО.

Сущность проекта (таблица «Project») носит информационный характер и используется для объединения в единую смысловую конструкцию нескольких расчетов. Расчеты (таблица «Calculation») являются основным способом взаимодействия с удаленным вычислителем, при создании которого задаются сценарий и его версия. В результате система генерирует графический интерфейс для ввода исходных данных, управления расчетными заданиями и доступа к выходным данным на удаленном вычислителе.

## Веб-сервер

Взаимодействие с сервером цифровой платформы основано на использовании графических и иных элементов пользовательского интерфейса, в результате взаимодействия с которыми веб-браузер отправляет на сервер соответствующий запрос. Маршрутизация запросов осуществляется посредством унифицированных указателей ресурса (URL) и имеет следующий вид:

- GET /auth/ - проверка актуальности сессии текущего пользователя;
- POST /auth/login/ - авторизация пользователя;
- POST /auth/logout/ - завершение сессии текущего пользователя;
- POST /auth/register/ - регистрация нового пользователя;

– GET /files/ - получение файла, расположенного в рабочей директории веб-сервера;

– GET /files/ssh/ - получение файла с удаленного вычислительного ресурса;

– POST /files/tmp/ - отправка пользовательского файла на сервер во временное хранилище;

– GET /api/ - при обращении открывает основное веб-сокет соединение, предназначенное для обеспечения взаимодействия между клиентом (веб-браузером) и сервером цифровой платформы;

– GET /terminal/ - при обращении открывает веб-сокет соединение с веб-браузером пользователя, а также устанавливает SSH-соединение с требуемым удаленным вычислителем. После успешной установки соединений сервер связывает веб-сокет с SSH-клиентом.

Как указано выше, основным каналом взаимодействия между веб-браузером пользователя и сервером цифровой платформы является веб-сокет, открываемый при обращении по URL-адресу /api/. Сообщения, отправляемые клиентом, должны содержать обязательное поле «url» для вызова корректного обработчика запроса и опциональное поле «data», содержащее добавочную информацию. Поле «url» должно представлять собой строку, во многом соответствующую стандартному URL-адресу. Ответ сервера отправляется в сообщении, содержащем поля «url», «body» и числового кода «status».

Необходимость наличия в ответе поля «url» объясняется особенностью работы веб-сокета, заключающейся в отсутствии классической схемы: запрос клиента и последующий ответ сервера. При использовании соединения данного типа клиент может только лишь подписаться на сообщения, отправляемые сервером (установка обработчика onmessage). В рамках разработанного подхода единый обработчик входящих сообщений вызывает конкретные клиентские функции, установленные при отправке запроса на сервер, на основании поля «url» ответа. Существует два типа устанавливаемых обработчиков – постоянные, при реагировании на сообщения о новом уведомлении или сообщении, подписке на изменение статуса расчетного задания, и единоразовые, используемые при запросе информации из базы данных, списка файлов на удаленном вычислителе.

Отметим, что масштабирование веб-сервера производится посредством создания подпроцессов средствами Node.js. При этом главный процесс обеспечивает задачи слежения за расчетными заданиями и состоянием вычислительных ресурсов. Процессы-потомки используются для обработки пользовательских запросов и обслуживания установленных веб-сокет соединений.

## Пользовательский веб-интерфейс

Для обзора клиентской части разработанного прототипа цифровой платформы рассмотрим типовые сценарии ее использования, в том числе

регистрацию вычислительных ресурсов, приложений, сценариев и проведения вычислительного эксперимента.

Доступ к цифровой платформе происходит по обращению пользователя через браузер его ПК по корневому URL-адресу. Установленное соединение приведет пользователя к форме авторизации, изображенной на рис. 3.

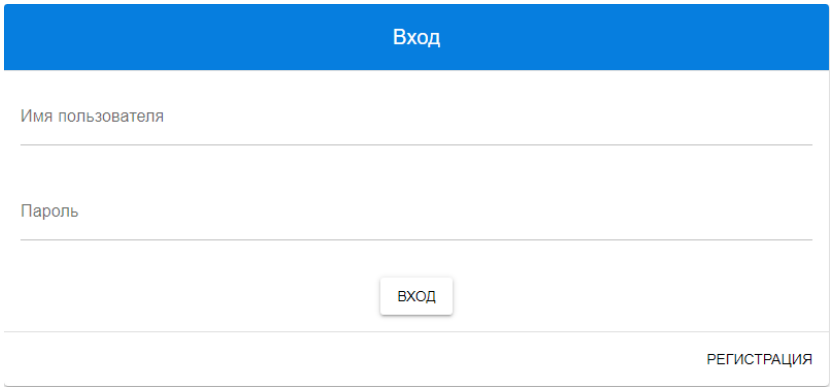


*Рис. 3.* Форма авторизации

При отсутствии аккаунта предполагается обращение пользователя к администратору системы с целью получения ключа доступа, генерируемого системой автоматически и определяющего тип аккаунта. При наличии ключа пользователь получает возможность пройти регистрацию посредством соответствующей формы (см. рис. 4).

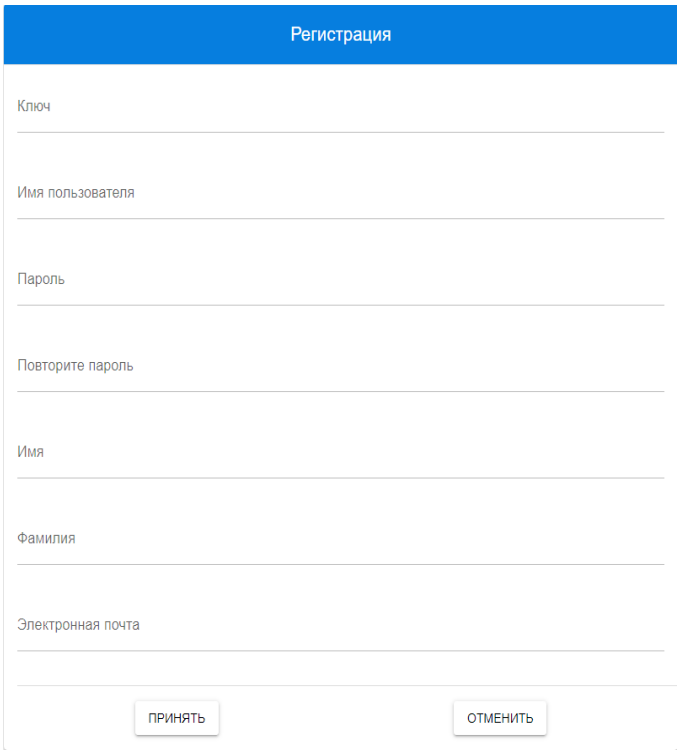


*Рис. 4.* Форма регистрации

После успешной авторизации веб-интерфейс принимает вид в соответствии с ролью пользователя. Интерфейс представляет собой набор корневых таблиц, отображающих списки основных сущностей – «Проекты», «Приложения», «Сценарии» и «Вычислители», доступ к которым осуществляется посредством панели навигации (см. рис. 5), а также специальные, соответствующие записи, страницы. Рассмотрим далее типовые сценарии использования системы для каждого из типов аккаунта. Отметим, что администратор системы имеет доступ к элементам управления, доступным для остальных ролей, а разработчику доступны возможности расчетчика.

Администратор системы отвечает за регистрацию новых вычислительных ресурсов, подготовку программного окружения и управление базой данных. При переходе на страницу «Вычислители», изображенную на рис. 5, администратор нажатием кнопки «Новая запись» получает доступ к форме регистрации вычислительного ресурса (см. рис. 6). К заполнению предлагаются следующие поля:

– «Имя» - название вычислительного ресурса, отображаемое в интерфейсе;
– «Тип доступа» - тип доступа к вычислительному ресурсу (в рамках прототипа – только SSH-соединение);
– «Адрес» - IP-адрес вычислительного ресурса или его доменное имя;
– «Порт» - порт для установления соединения;
– «Система управления очередью» - система управления заданиями пользователей, используемая на регистрируемом вычислительном ресурсе. Возможен выбор из двух вариантов – «native» (стандартный запуск приложений UNIX) и «SUPPZ» (СУППЗ);
– «Базовое окружение» - список переменных окружения.

После заполнения предложенной формы для регистрации вычислительного ресурса в системе необходимо нажать кнопку «Принять».

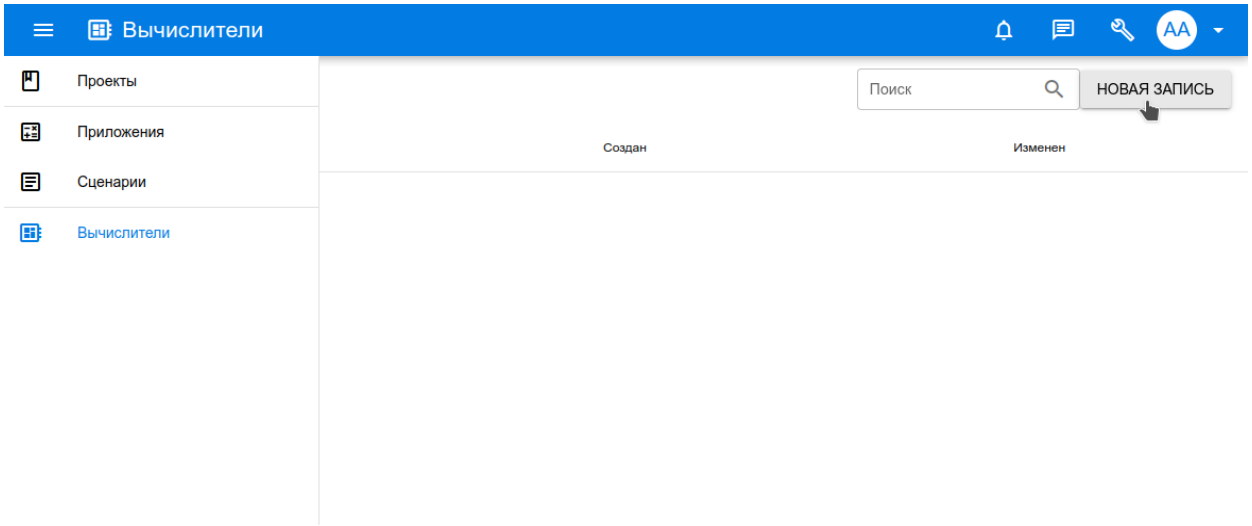


*Рис. 5.* Страница списка вычислительных ресурсов

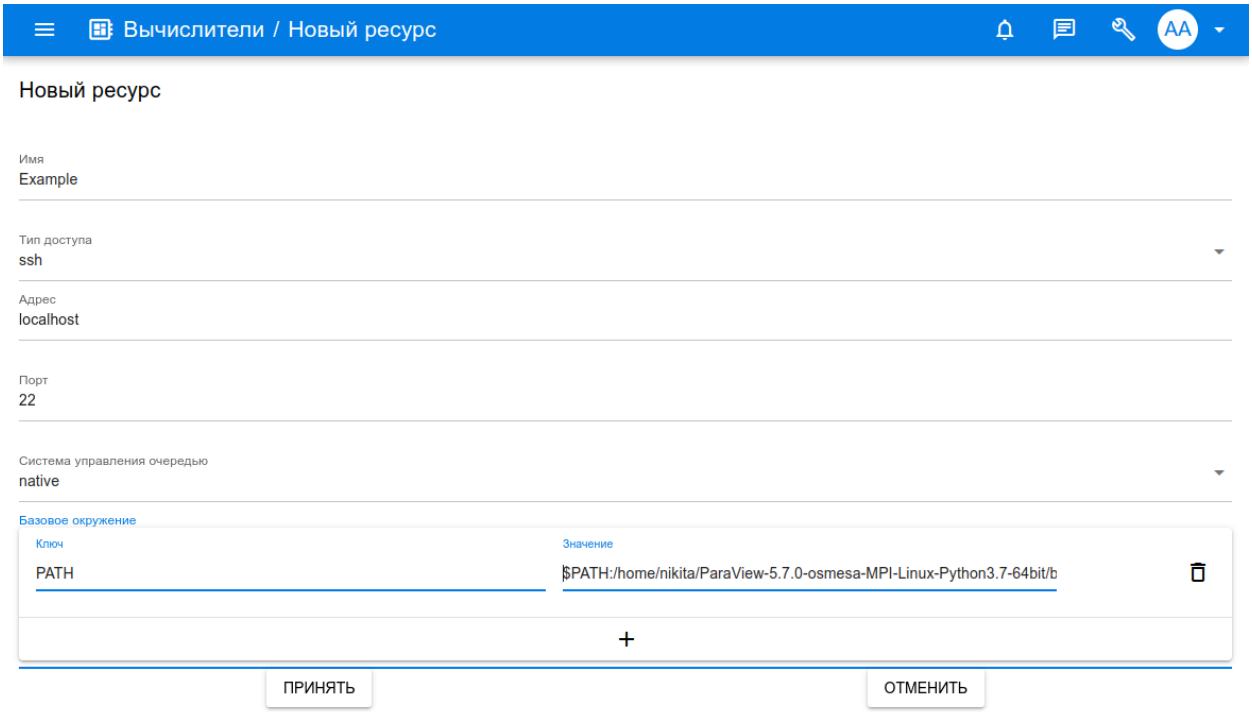


*Рис. 6.* Форма регистрации нового вычислительного ресурса

При создании записи система отобразит страницу вычислительного ресурса, изображенную на рис. 7. В рамках разработанного прототипа авторизация пользователя на удаленном вычислителе осуществляется с помощью RSA-ключа сервера, который необходимо добавить в соответсвующий файл на удаленном вычислителе (в UNIX-системах обычно это файл authorized_keys, расположенный в .ssh, находящейся в домашней директории). Получить данный ключ возможно посредством нажатия кнопки «Показать SSH-RSA».

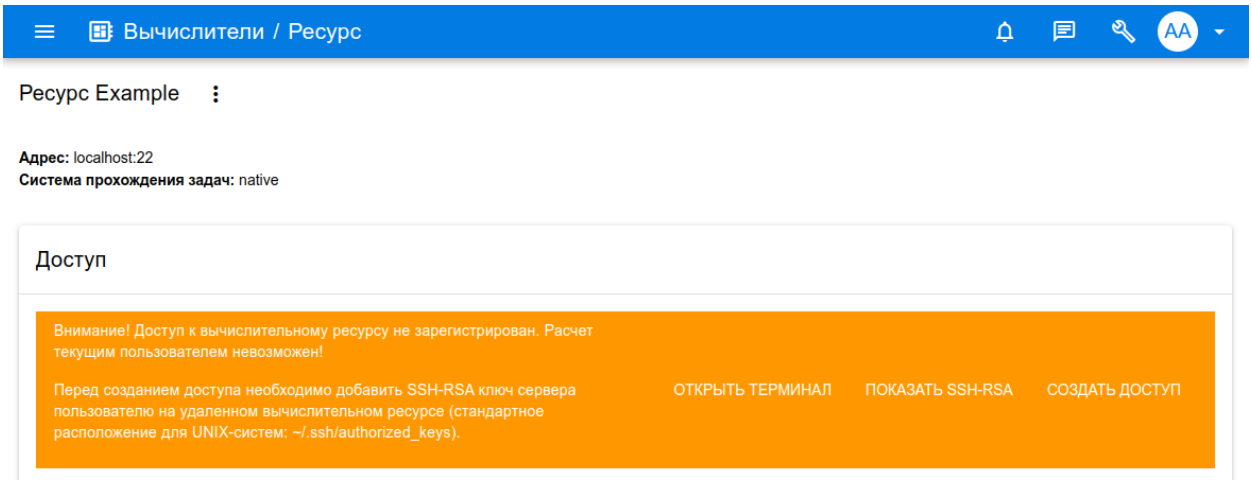


*Рис. 7.* Страница вычислительного ресурса

После осуществления подготовки UNIX-пользователя на удаленном вычислителе необходимо нажать кнопку «Создать доступ», в результате чего в интерфейсе отобразится форма создания доступа, изображенная на рис. 8. Поля,

подлежащие заполнению, включают в себя «Имя пользователя» (имя пользователя на удаленном вычислителе) и «Рабочая директория» (директория, в которой будут храниться данные проектов и расчетов), а также дополнительный список переменных окружения.

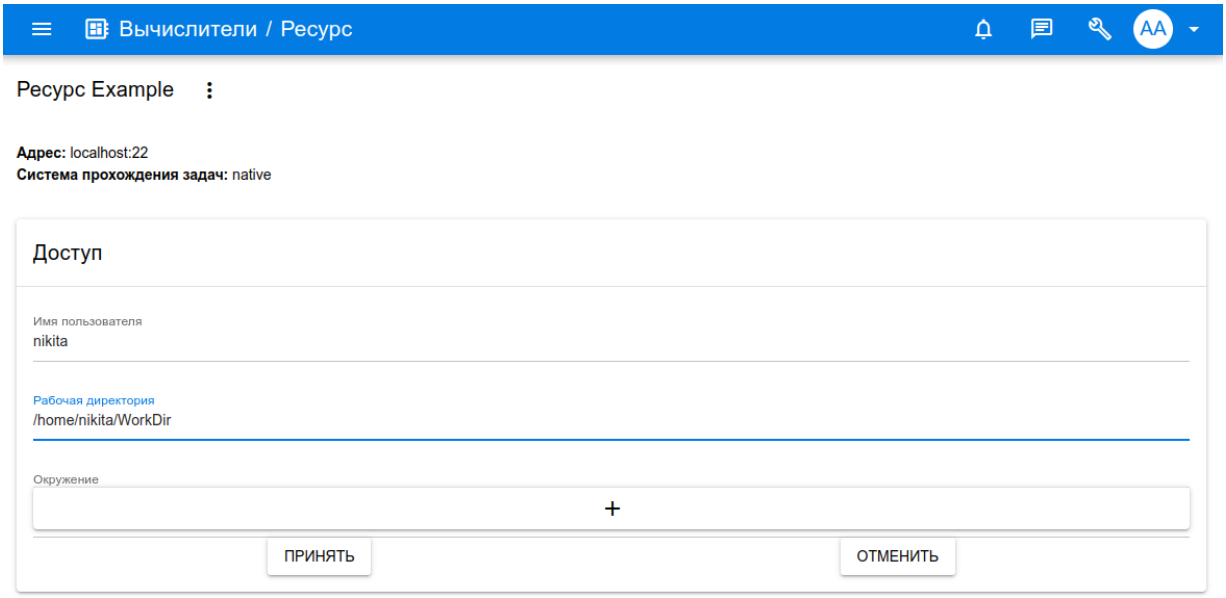


*Рис. 8.* Форма создания доступа

После завершения заполнения формы нажатием кнопки «Принять» сервер проверит возможность соединения и авторизации с удаленным вычислительным ресурсом. При успешном соединении в таблице «Access» будет создана новая запись. Страница примет вид, подобный изображенному на рис. 9.

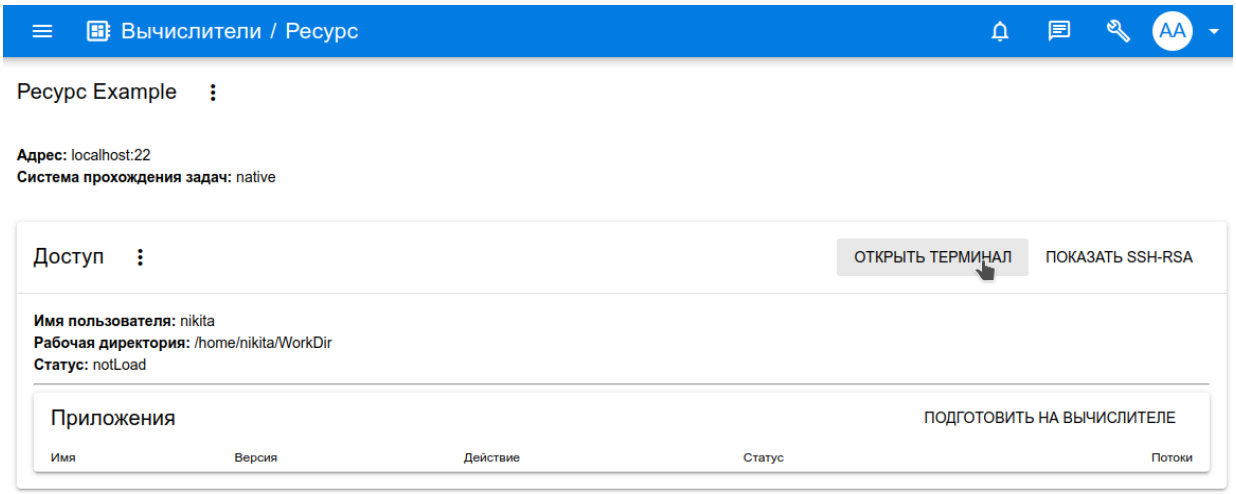


*Рис. 9.* Страница вычислительного ресурса с активным доступом

Кроме того, реализована возможность прямого взаимодействия с удаленным вычислителем посредством SSH-терминала (см. рис. 10), для чего достаточно нажать кнопку «Открыть терминал». Отметим, что перед

использованием удаленного ресурса для проведения вычислительного эксперимента пользователь любой роли должен предварительно подготовить доступ и окружение.

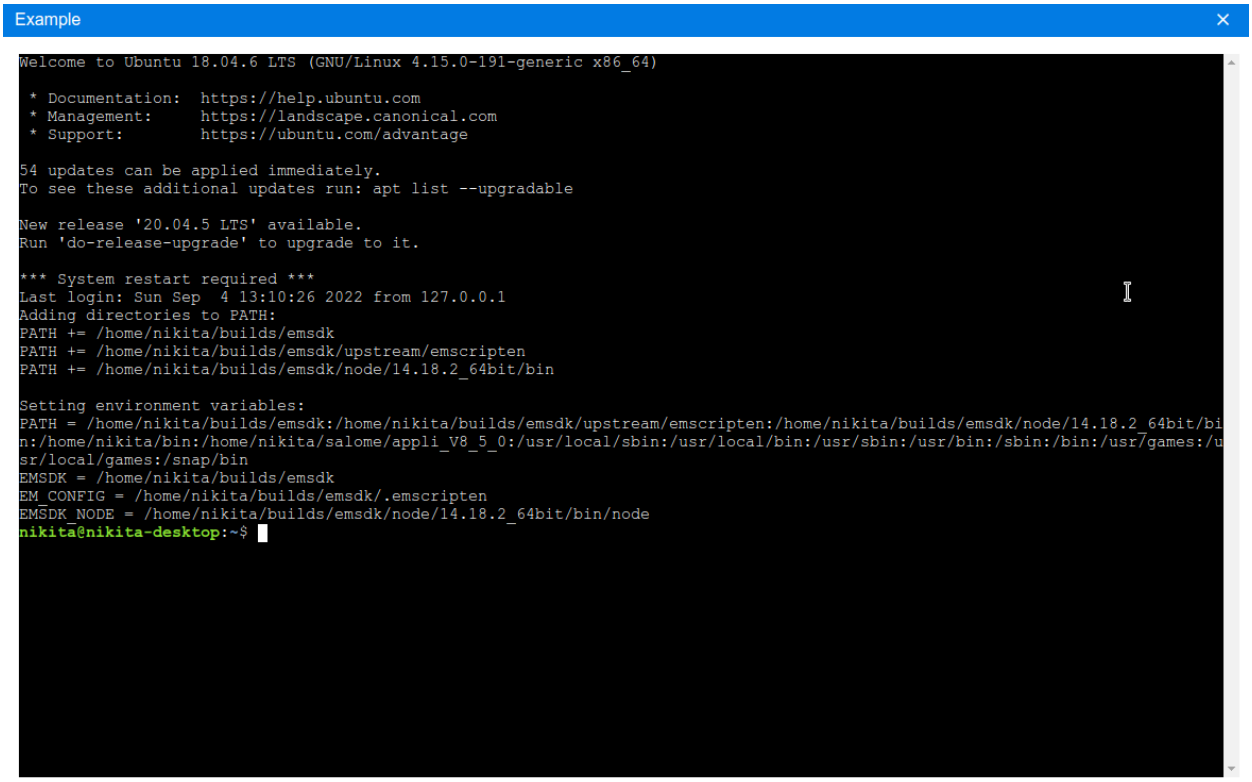


*Рис. 10.* Терминал вычислительного ресурса

Администратор системы, кроме возможности регистрации в системе новых вычислительных ресурсов, имеет возможность осуществить прямой доступ к таблицам базы данных посредством страницы, изображенной на рис. 11. Посредством предлагаемого интерфейса пользователь может просмотреть, изменить или удалить записи из базы данных.

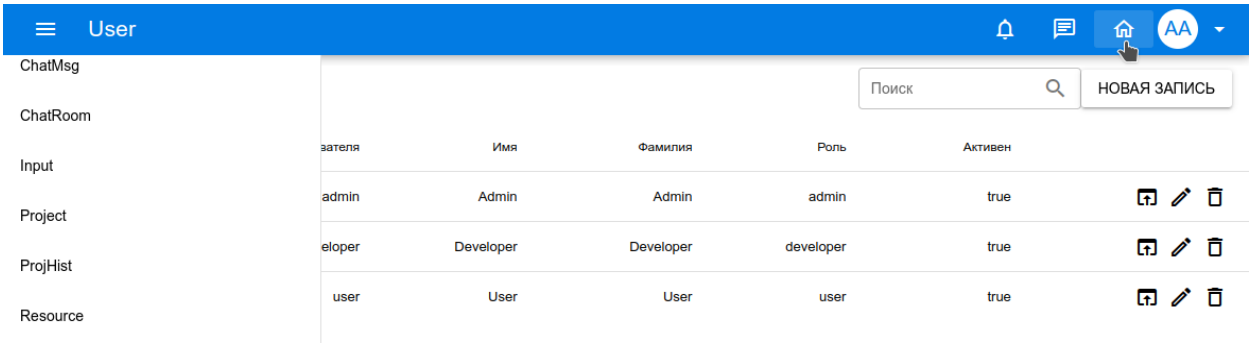


*Рис. 11.* Интерфейс доступа к базе данных

Основной областью деятельности разработчика прикладного програмного обеспечения являются формы регистрации приложений и сценариев на

цифровой платформе. Обращаясь на страницу приложений, представленную на рис. 12, разработчик может инициировать процесс создания новой записи нажатием кнопки «Новая запись». После этого система отобразит форму создания нового приложения, изображенную на рис. 13. Полями, подлежащими заполнению, являются:

– «Имя» - название приложения, отображаемое в интерфейсе пользователя;
– «Идентификатор» - уникальный идентификатор, посредством которого производится обращение к приложению в сценарии запуска;
– «Публичный» - флаг, указывающий возможность доступа к приложению других пользователей системы;
– «Описание» - текстовое описание приложения, поддерживающее разметку в формате Markdown [35];
– «Шаблон паспорта» - шаблон функционального описания приложения, подставляется как значение по умолчанию в форме создания новых версий приложения.

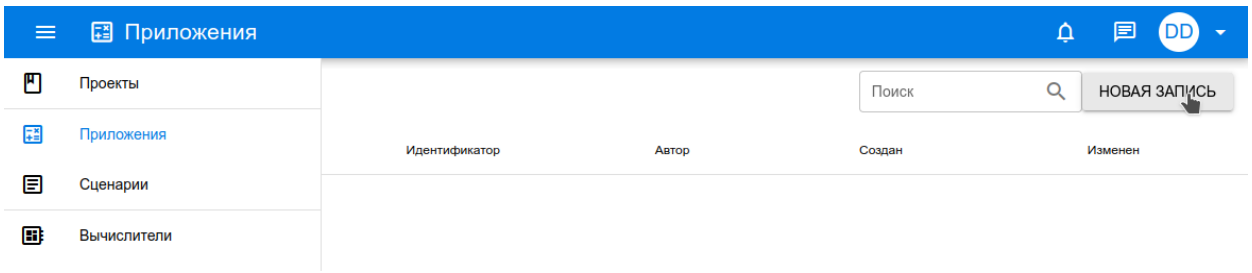


*Рис. 12.* Страница приложений

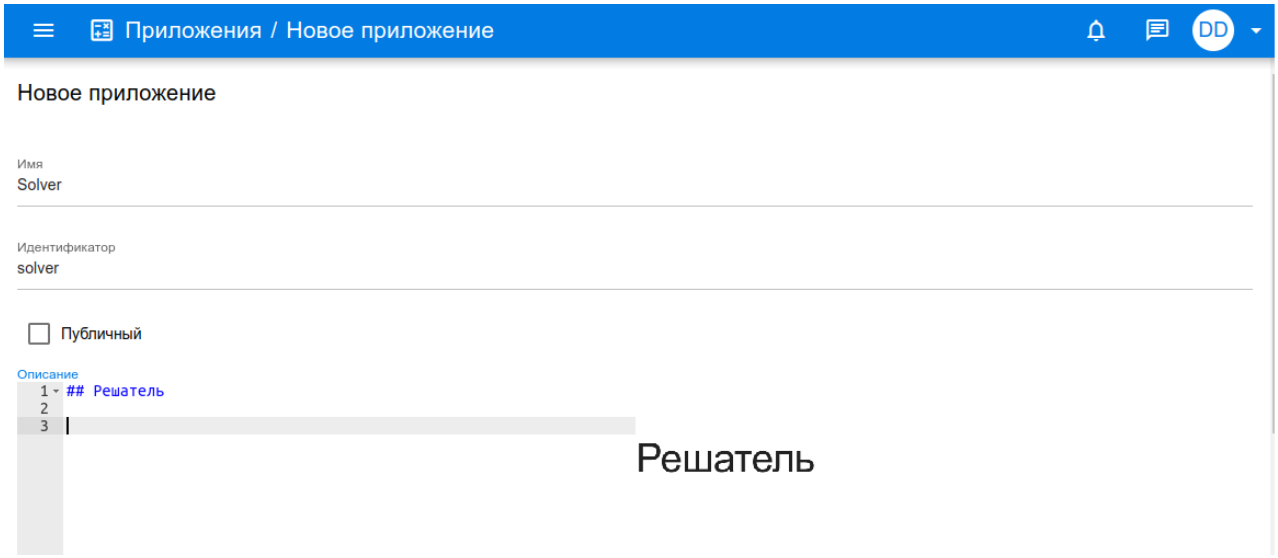


*Рис. 13.* Форма создания приложения

Нажатие кнопки «Принять» в форме отправит предоставленные пользователем данные на веб-сервер, который создаст новую запись в таблице базы данных, после чего пользователь будет перенаправлен на страницу приложения подобную, изображенной на рис. 14. Нажатие кнопки «Новая запись» в таблице «Версии» перенаправит разработчика к форме создания новой версии приложения. Полями, подлежащими заполнению, являются:

– «Имя» - название версии приложения, используемое для отображения в интерфейсе;
– «Описание» - текстовое описание версии приложения;
– «Паспорт» - функциональное описание приложения, текстовый файл в формате YAML [36], содержащий такие элементы, как команды сборки и запуска, списки входных и выходных данных, их описание;
– «Исходный код» - архив, содержащий исходный код описываемой версии приложения.

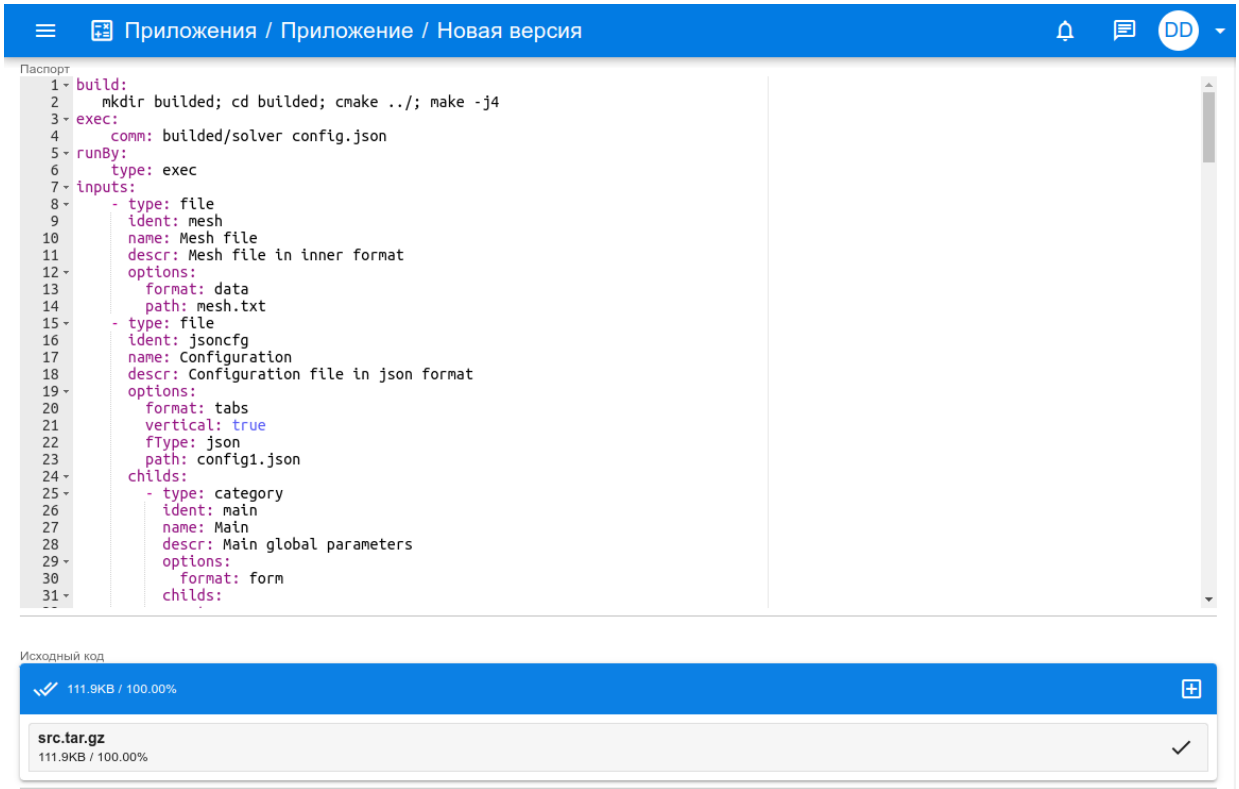


*Рис. 14.* Форма создания версии приложения

Для подготовки приложения на удаленном вычислительном ресурсе необходимо активировать вкладку «Подготовка», нажатие на которую отобразит таблицу «Ресурсы», изображенную на рис. 15 и содержащую список вычислительных ресурсов, зарегистрированных в системе. В столбце версия пользователем определяется версия для подготовки. Последующее нажатие кнопки «Подготовить на вычислителях» запустит процесс отправки исходного кода на удаленный вычислительный ресурс с последующей сборкой

исполняемого файла. Для доступа к стандартным потокам пользователю достаточно нажать кнопки «Stdout» или «Stderr». В процессе подготовки приложения столбцы «Действие» и «Статус» будут содержать описание текущего этапа и его состояние.

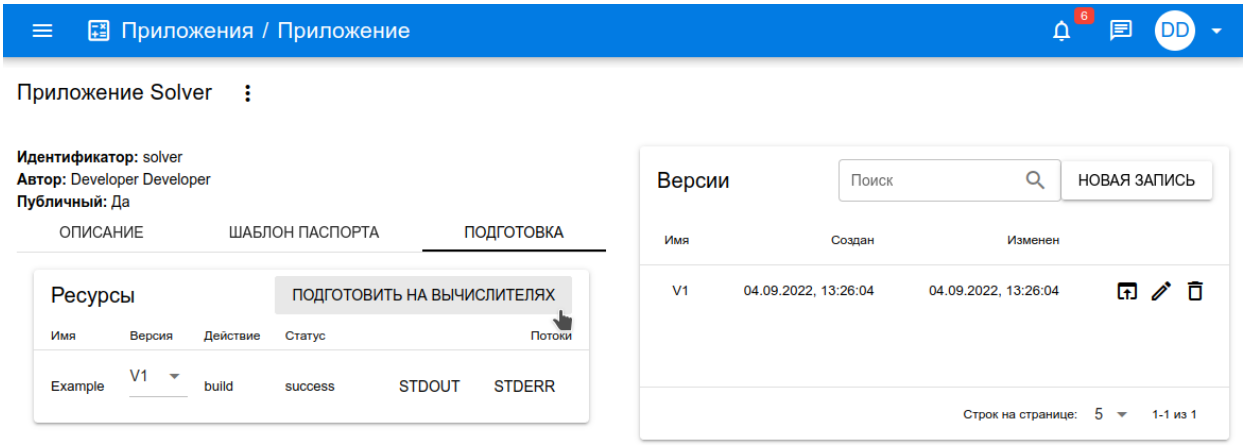


*Рис. 15.* Страница приложения. Подготовка на удаленном вычислительном ресурсе

Для создания сценариев на соответствующей странице, содержащей их список, необходимо нажать кнопку «Новая запись». Система предложит разработчику заполнить форму, изображенную на рис. 16 и содержащую поля:

– «Имя» - название сценария, отображаемое в системе;
– «Публичный» - флаг, отмечающий доступность сценария для других пользователей;
– «Описание» - текстовое описание сценария, поддерживающее разметку Markdown;
– «Шаблон скрипта» - используется для подстановки в поле «Скрипт» версии сценария как значение по умолчанию.

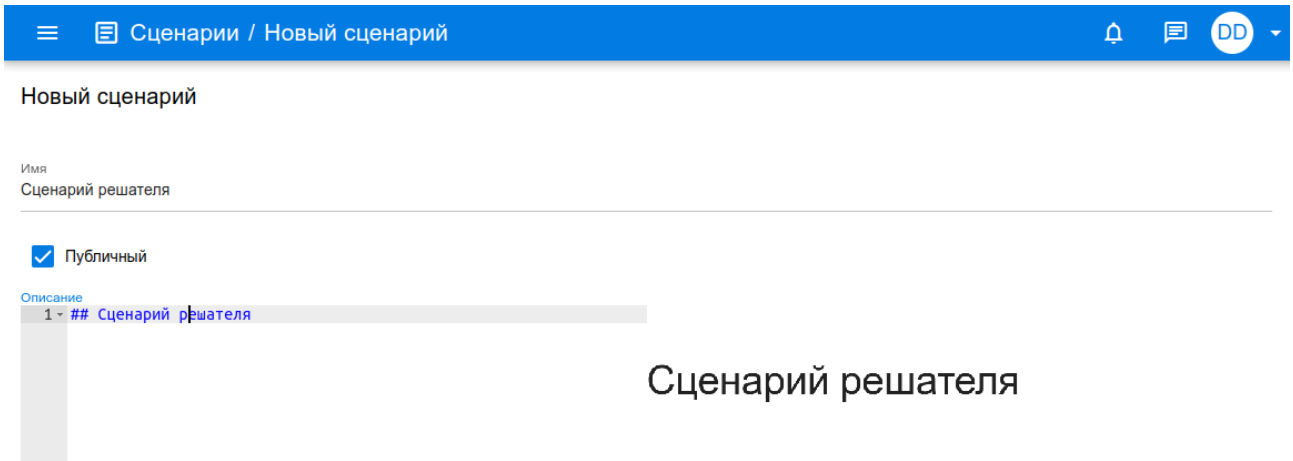


*Рис. 16.* Форма создания сценария

После окончания работы с формой необходимо нажать кнопку «Принять» для завершения ввода. В результате отправки данных на сервер и создания записи пользователь будет перенаправлен на страницу сценария. Нажатие кнопки «Новая запись» в таблице «Версии» перенаправит разработчика к форме создания новой версии сценария, изображенной на рис. 17. Данная форма содержит поля:

– «Имя» - название версии сценария;
– «Описание» - текстовое описание версии сценария в формате Markdown;
– «Скрипт» - функциональное описание сценария в формате YAML, содержащее ключи: «applications» - словарь приложений, где в качестве ключа используется строка, обозначающая приложение в рамках сценария, а в качестве значения – идентификатор приложения; «connections» - связи входных и выходных данных приложений; «stages» - этапы запуска приложений.

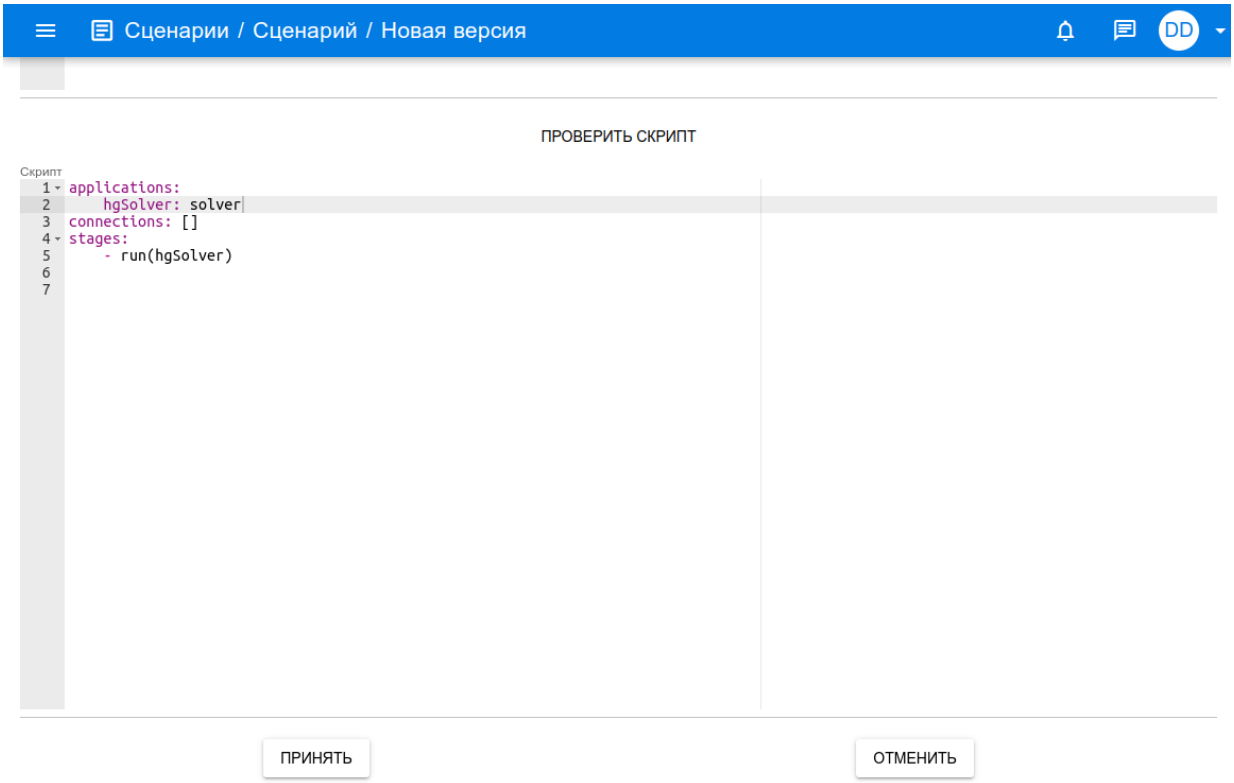


*Рис. 17.* Форма создания версии сценария

Нажатие кнопки «Проверить скрипт» отобразит шаблон расчета, изображенный на рис. 18, позволяющий проверить корректность YAML-файла, существование приложений с указанными идентификаторами в базе данных, корректность форм входных и выходных данных. После окончания работы с формой для подтверждения ввода и отправки данных на сервер достаточно нажать кнопку «Принять».

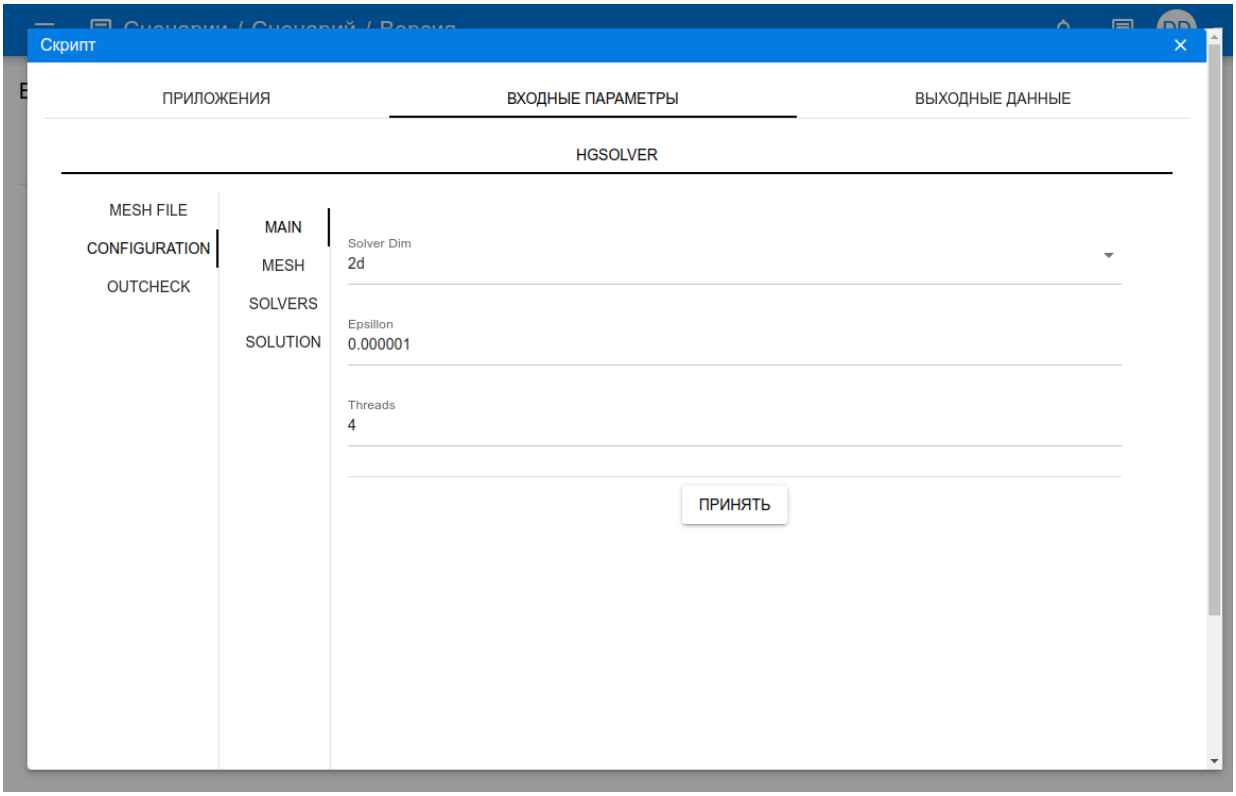


*Рис. 18.* Проверка корректности скрипта

Основной задачей пользователя с ролью расчетчик является проведение вычислительных экспериментов. В рамках разработанного прототипа данный процесс реализуется следующим образом. Нажатие на кнопку «Новая запись» на странице, содержащей список проектов, направит пользователя на страницу создания проекта, имеющую форму с полями (см. рис. 19):

– «Имя» - название проекта;
– «Описание» - текстовое описание проекта в формате Markdown;
– «Участники» - список пользователей – участников проекта.

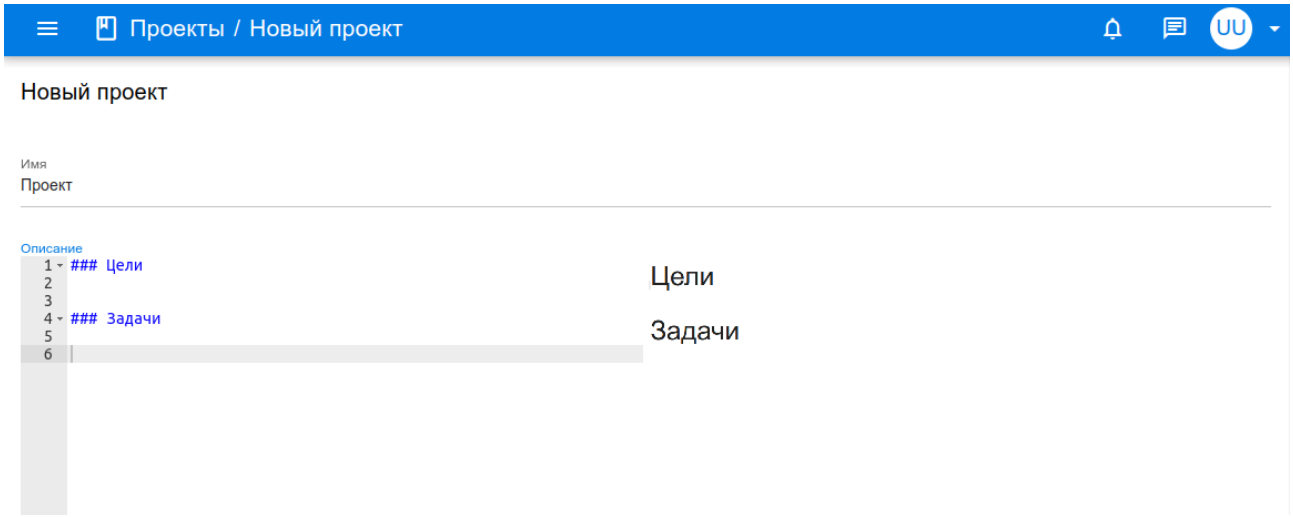


*Рис. 19.* Форма создания проекта

После заполнения предложенных полей для передачи информации на сервер и создания записи в базе данных необходимо нажать кнопку «Принять». В результате система предложит страницу проекта, подобную изображенной на рис. 20.

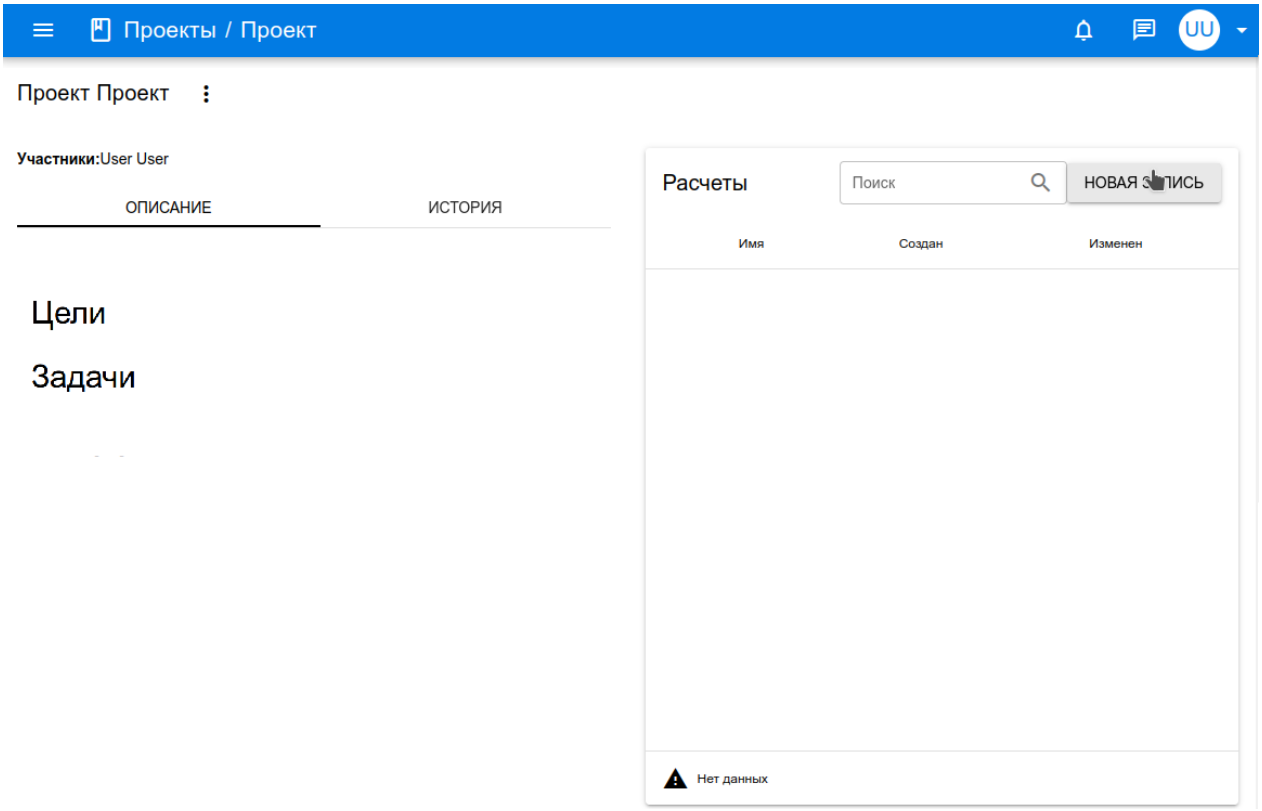


*Рис. 20.* Страница проекта

Для создания нового расчета в таблице «Расчеты» необходимо нажать кнопку «Новая запись». Система перенаправит пользователя на страницу, содержащую форму создания нового расчета, изображенную на рис. 21. Данная форма содержит поля:

- «Имя» - название расчета;
- «Описание» - текстовое описание расчета в формате Markdown;
- «Сценарий» и «Версия» - выпадающие списки для выбора сценария и его версии.

После завершения работы с формой для регистрации в системе нового расчета достаточно нажать кнопку «Принять». В результате создания записи пользователь будет направлен на страницу, подобную изображенной на рис. 22. На странице расчета происходит основное взаимодействие с удаленным вычислительным ресурсом, включающее управление и слежение за пользовательскими процессами, подготовку исходных данных и анализ выходных. На вкладке «Вычислители» пользователь может определить, на каком из вычислительных ресурсов будет происходить запуск приложения из списка сценария.

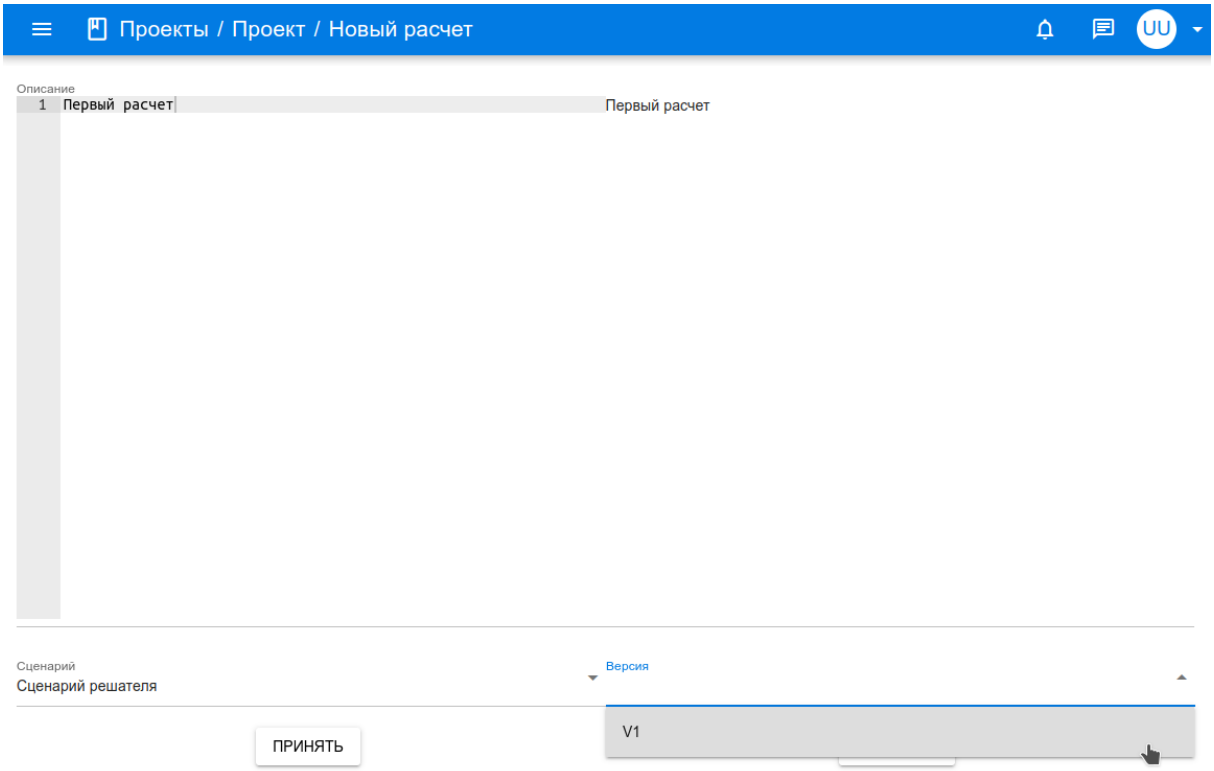


*Рис. 21.* Форма создания расчета

Активация «Входные параметры» отобразит формы в соответствии с описанием, указанным в паспорте приложения. При условии, что входным параметром являются файл данных или директория, системой будет отрисована форма, изображенная на рис. 22. Посредством данной формы возможно либо отправить файл, либо связать с результатами расчетов, полученными в системе ранее.

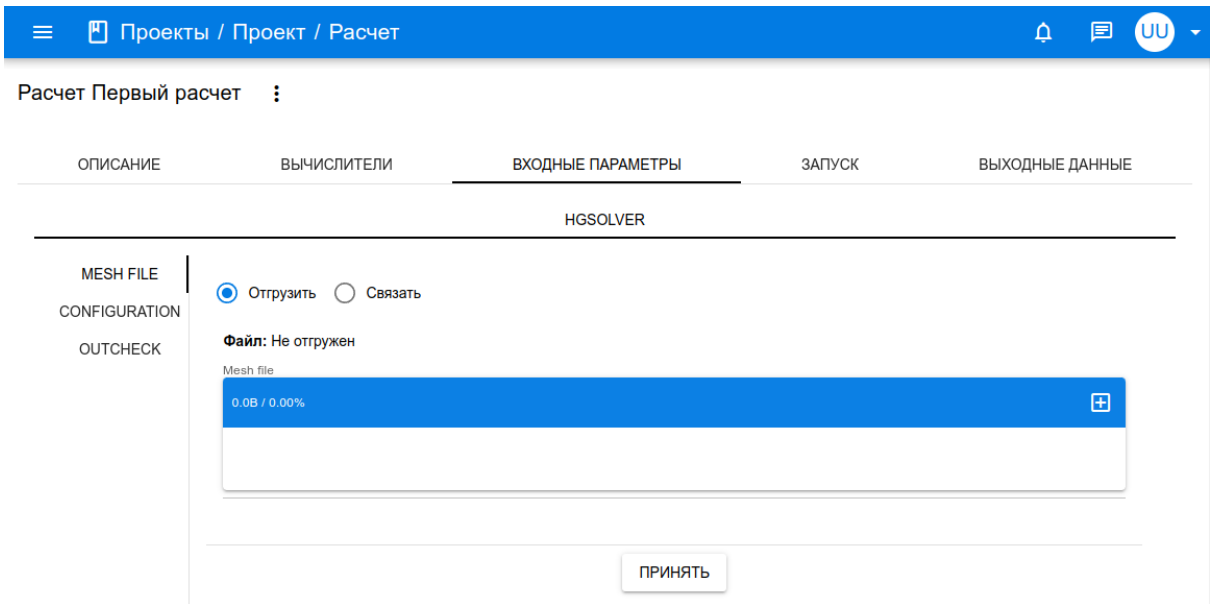


*Рис. 22.* Задание входных данных

Если параметры являются числовыми, строковыми, списками или иными поддерживаемыми типами, оснащенными компонентами, система предложит комплексную веб-форму. Пример подобного компонента изображен на рис. 23. Подтверждение ввода производится нажатием кнопки «Принять».

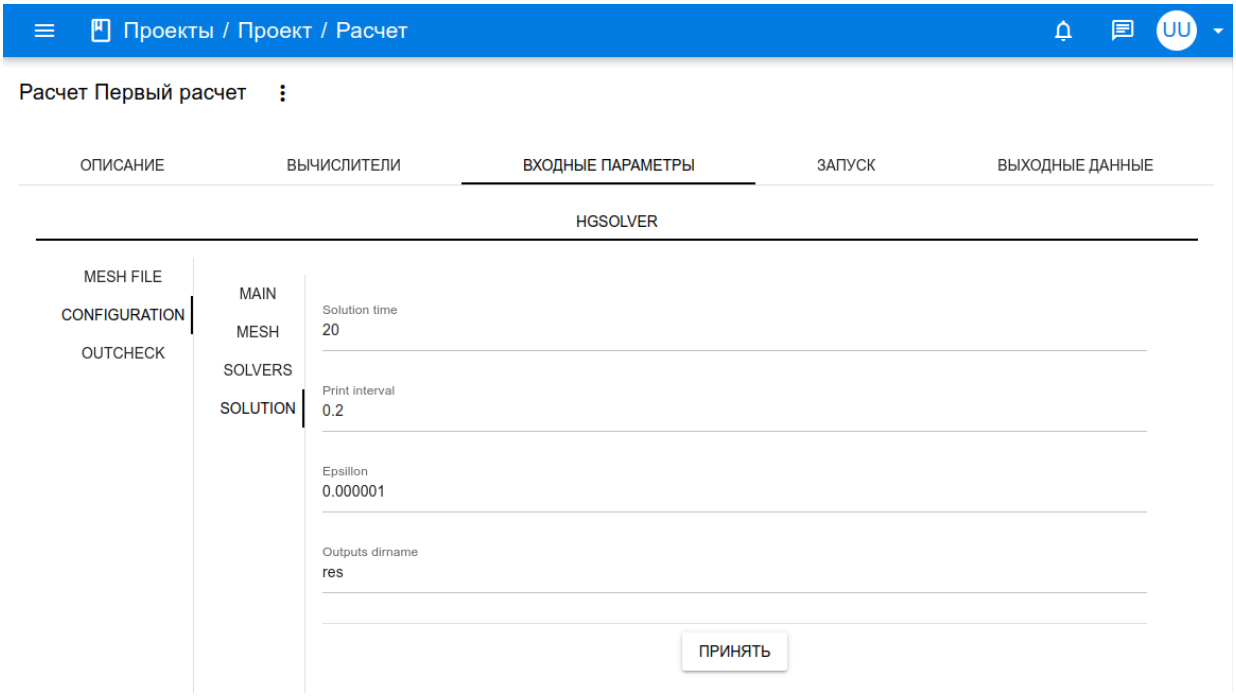


*Рис. 23.* Задание параметров расчета

После завершения подготовки входных параметров возможно произвести запуск задачи на расчет. Активация вкладки «Запуск» представит компонент управления расчетным процессом (см. рис. 24). В таблице «Приложения» в зависимости от типа запуска и системы прохождения пользовательских заданий столбец «Параметры» будет содержать необходимые для запуска поля (например –np для процессов, запускаемых с помощью mpirun).

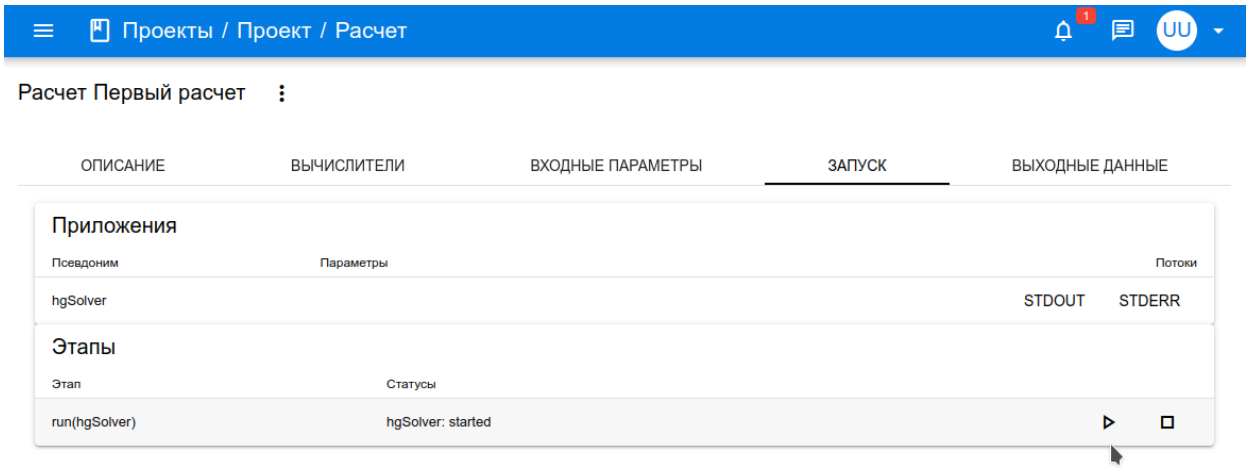


*Рис. 24.* Компонент управления расчетным заданием

Управление (запуск, останов) приложениями осуществляется посредством таблицы «Этапы» с помощью нажатия соответствующих кнопок. При этом актуальный статус задачи будет указан в столбце «Статусы». Для просмотра стандартных потоков вывода и ошибки достаточно нажать кнопки «Stdout» и «Stderr» соответственно (см., например, рис. 25).

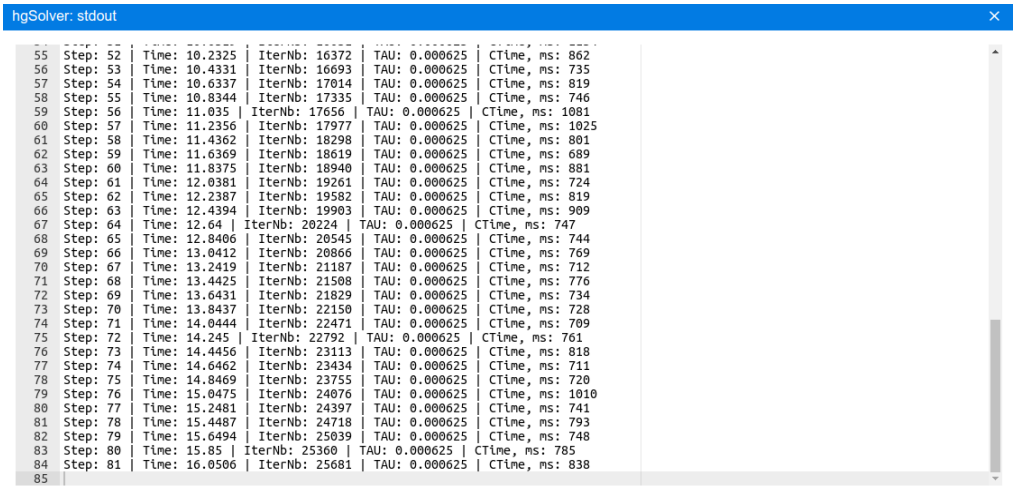


*Рис. 25.* Просмотр стандартного потока вывода

Для анализа выходных данных пользователю необходимо активировать вкладку «Выходные данные», в результате чего система представит компонент пользовательского интерфейса, подобный изображенному на рис. 26. В левой части компонента будет сформирована древовидная структура, содержащая списки выходных данных. Для директорий предусмотрена возможность отображения списка содержащихся в них файлов. При активации элемента списка справа будет отображаться информация, включая описание, время изменения, размер и тип. Также пользователь может воспользоваться такими функциями, как «Открыть», «Скачать» или «Очистить».

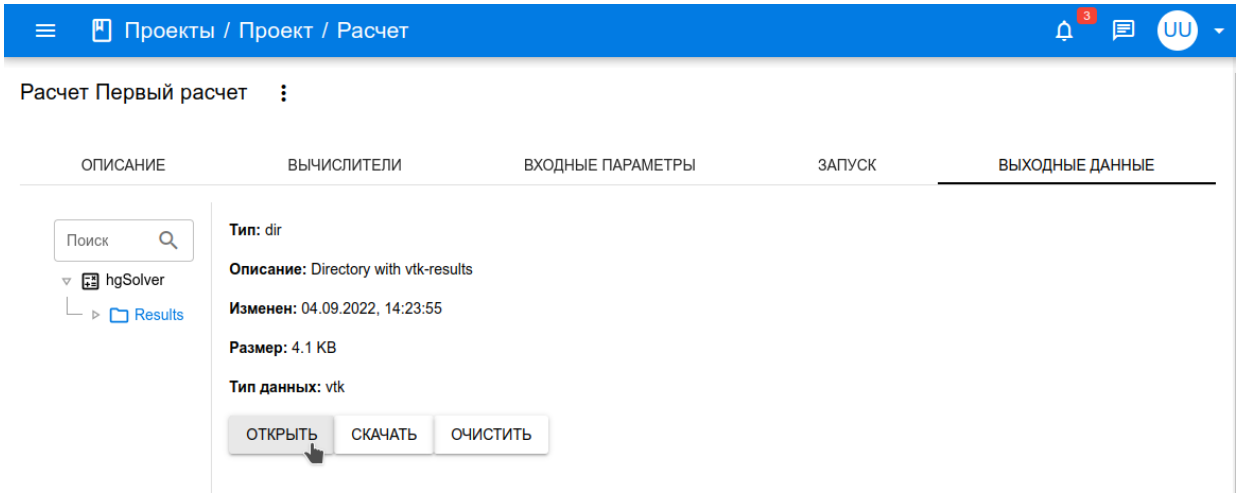


*Рис. 26.* Список выходных данных

Поведение системы при нажатии кнопки «Открыть» зависит от типа активных выходных данных. Например, при открытии типа «vtk» система отобразит компонент, содержащий Visualizer.js, пример которого изображен на рис. 27. В рамках прототипа реализованы возможности непосредственного просмотра растровых изображений и видеозаписей, текстовых файлов и данных библиотеки VTK.

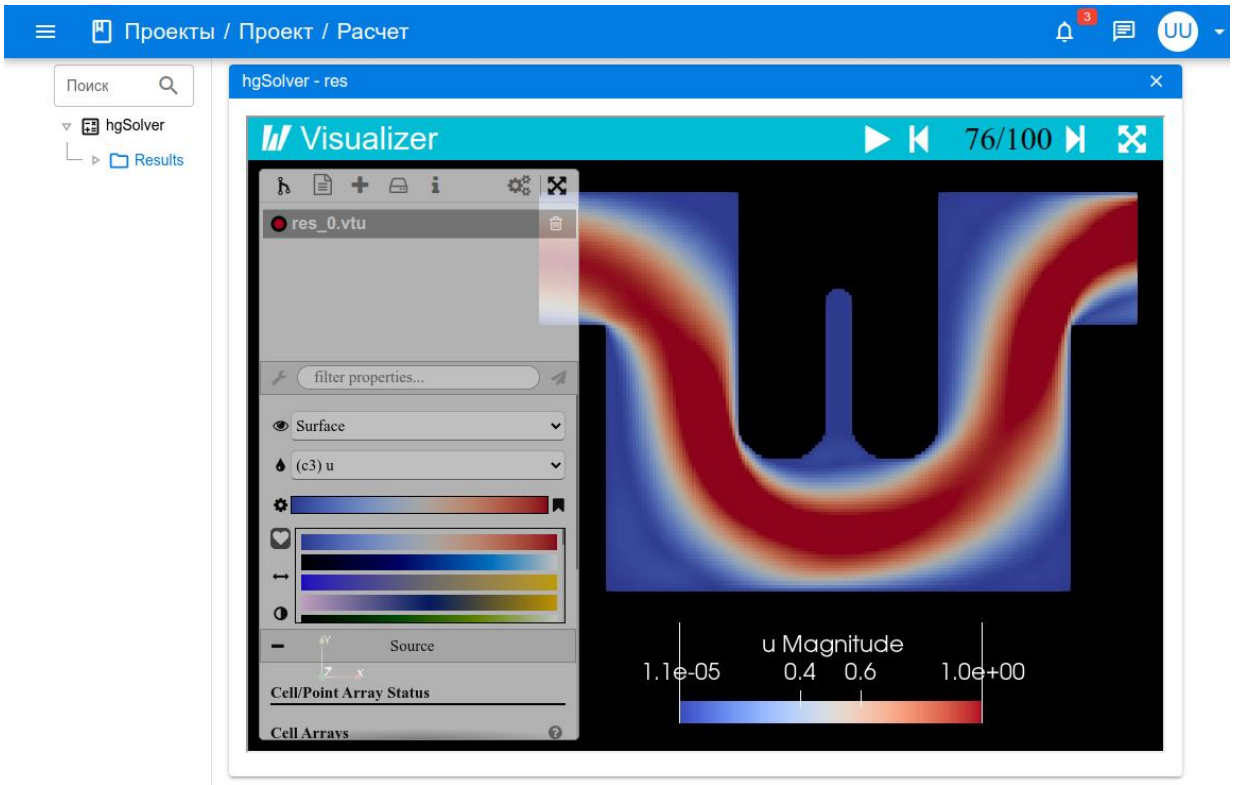


*Рис. 27.* Просмотр файлов визуализации

Для демонстрации возможностей трехмерной интерактивной визуализации, осуществляемой непосредственно в веб-браузере пользователя, приведем в качестве примера задачу о течении жесткой воды через область сложной геометрии. В центре области расположен нагревательный элемент, течение жидкости осуществляется по направлению оси X. В результате численного расчета описанной задачи на тетраэдральной сетке был получен набор файлов данных в формате VTK для различных моментов времени. Каждый файл содержит сеточное представление, а также списки моделируемых величин, отнесенных к сеточному элементу. Визуализация модуля скорости представлена на рис. 28, температура и давление изображены на рис. 29, а распределение концентрации на рис. 30.

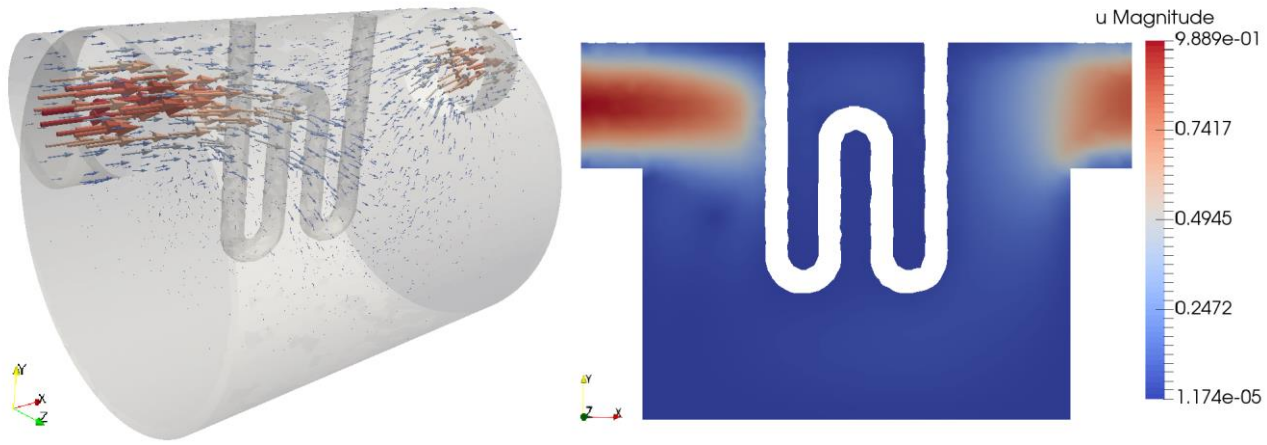


*Рис. 28.* Распределение модуля скорости в объеме (слева) и сечении Z=0 (справа)

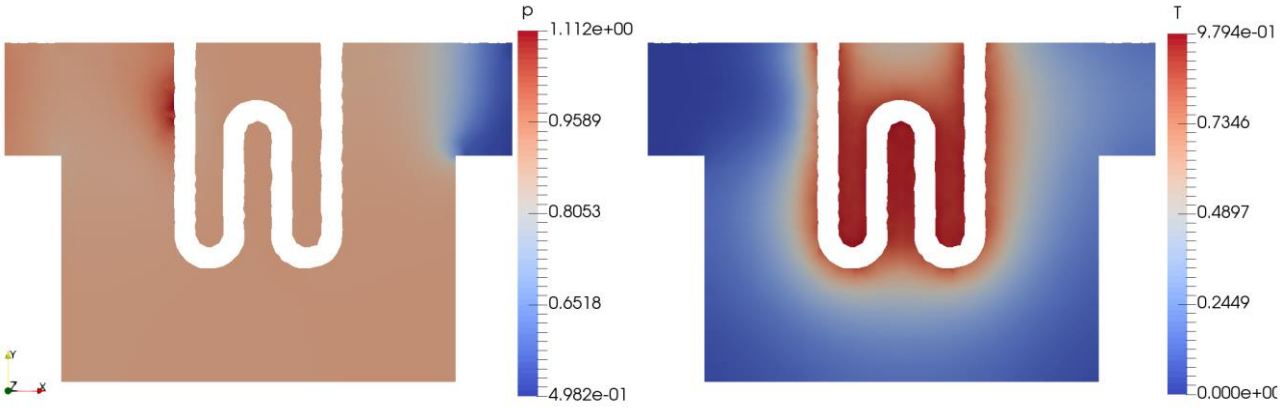


*Рис. 29.* Распределения давления (слева) и температуры (справа)

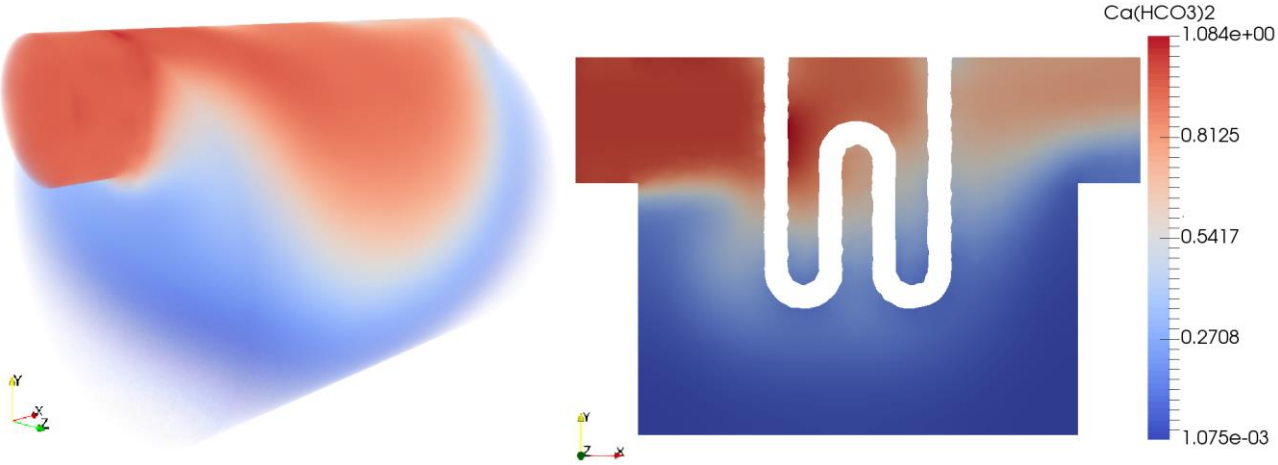


Рис. 30. Распределение концентрации соли в объеме (слева) и в сечении Z=0 (справа)

## Заключение

В работе рассмотрен принцип построения цифровой платформы, предназначенной для проведения комплексных численных расчетов. Основным назначением разработанного прототипа системы являются упрощение и унификация взаимодействия с удаленными вычислительными ресурсами. Предложенная реализация может существенно облегчить подготовку исходных кодов и сборку ПО, позволяя наладить непрерывное взаимодействие между разработчиком прикладной программы и ее конечным пользователем. Для проведения вычислительных экспериментов предусмотрены возможности генерации графического интерфейса для задания параметров и данных на основе файла-описания. Система управления и мониторинга позволяет осуществлять запуск и слежение за пользовательскими заданиями на разных вычислительных ресурсах в рамках единого веб-решения. Анализ выходных данных предлагается осуществлять с помощью серверной отрисовки средствами свободного ПО ParaViewWeb.

## Библиографический список


1. Поляков С. В., Карамзин Ю. Н., Косолапов О. А., Кудряшова Т. А., Суков С. А. Гибридная суперкомпьютерная платформа и разработка приложений для решения задач механики сплошной среды сеточными методами // Известия ЮФУ. Технические науки. 2012. №6. С. 105-115.
2. ЦКП ИПМ РАН [Электронный ресурс]. URL: https://ckp.kiam.ru/?home
3. МВС-1000:: Документация :: Руководство пользователя системы МВС-1000/K-10 [Электронный ресурс]. URL: https://www.kiam.ru/MVS/documents/k60/userguide.html
4. Slurm Workload Manager - Documentation [Электронный ресурс]. URL: https://slurm.schedmd.com/documentation.html
5. Home – Docker [Электронный ресурс]. URL: https://www.docker.com/
6. Kubernetes Production-Grade Container Orchestration [Электронный ресурс]. URL: https://kubernetes.io/
7. Службы облачных вычислений | Microsoft Azure [Электронный ресурс]. URL: https://azure.microsoft.com/ru-ru/
8. Сервисы облачных вычислений – Amazon Web Services (AWS) [Электронный ресурс]. URL: https://aws.amazon.com/ru/
9. Надежное облако для вашего бизнеса — Yandex Cloud [Электронный ресурс]. URL: https://cloud.yandex.ru/
10. Поляков С. В., Подрыга В. О., Пузырьков Д. В. Облачный сервис для решения перспективных задач нанотехнологии // Научный сервис в сети Интернет: труды XIX Всероссийской научной конференции (18-23 сентября 2017 г., г. Новороссийск). М.: ИПМ им. М. В. Келдыша, 2017. С. 379-388. URL: http://keldysh.ru/abrau/2017/54.pdf

11. Пузырьков Д. В., Подрыга В. О., Поляков С. В. Облачный сервис для масштабных молекулярно-динамических расчетов: от идеи до реализации // Научный сервис в сети Интернет: труды XIX Всероссийской научной конференции (18-23 сентября 2017 г., г. Новороссийск). М.: ИПМ им. М.В.Келдыша, 2017. С. 406-416. URL: http://keldysh.ru/abrau/2017/58.pdf
12. Sim Streamlined For Design: structural simulation and analysis in the cloud [Электронный ресурс]. URL: https://www.sim4design.com/index
13. Nucleonica [Электронный ресурс]. URL: https://www.nucleonica.com/
14. Everest [Электронный ресурс]. URL: http://everest.distcomp.org/
15. Sukhoroslov O., Volkov S., Afanasiev A. A Web-Based Platform for Publication and Distributed Execution of Computing Applications // 14th International Symposium on Parallel and Distributed Computing (ISPDC). IEEE, 2015. pp. 175-184.
16. Puzyrkov, D.V., Podryga, V.O., Polyakov, S.V.: Cloud service for HPC management: ideas and appliance. Lobachevskii J. Math. 39(9), 2018. pp. 1251–1261. URL: https://doi.org/10.1134/S1995080218090172
17. Пузырьков Д.В., Поляков С.В., Тарасов Н.И. Программная среда для решения задач управления численными расчетами на суперкомпьютерах KIAM_WMCS, версия 1. Свидетельство о государственной регистрации программ для ЭВМ № 2018666098, 12 декабря 2018. Правообладатель: ИПМ им. М.В.Келдыша РАН.
18. Поляков С.В., Подрыга В.О., Пузырьков Д.В., Тарасов Н.И. Вэб-интерфейс пользователя для моделирования на суперкомпьютерах свойств молекулярных систем KIAM_MMD_WUI, версия 1. Свидетельство о государственной регистрации программ для ЭВМ № 2020661866, 01 октября 2020. Правообладатель: ИПМ им. М.В.Келдыша РАН.
19. The web framework for perfectionists with deadlines | Django. [Электронный ресурс]. URL: https://www.djangoproject.com/
20. Vue.js - The Progressive JavaScript Framework | Vue.js [Электронный ресурс]. URL: https://vuejs.org/
21. Node.js [Электронный ресурс]. URL: https://nodejs.org/en/
22. Express - Node.js web application framework [Электронный ресурс]. URL: https://expressjs.com/
23. Passport.js [Электронный ресурс]. URL: https://www.passportjs.org/
24. Sequelize | Feature-rich ORM for modern TypeScript & JavaScript [Электронный ресурс]. URL: https://sequelize.org/
25. SQLite Home Page [Электронный ресурс]. URL: https://sqlite.org/index.html
26. RFC 6455 - The WebSocket Protocol [Электронный ресурс]. URL: https://datatracker.ietf.org/doc/html/rfc6455
27. ws: a Node.js WebSocket library [Электронный ресурс]. URL: https://github.com/websockets/ws
28. nginx [Электронный ресурс]. URL: https://nginx.org/
29. Quasar Framework [Электронный ресурс]. URL: https://quasar.dev/

30. Home | Vue Router [Электронный ресурс]. URL: https://router.vuejs.org/
31. What is Vuex? | Vuex [Электронный ресурс]. URL: https://vuex.vuejs.org/
32. ParaViewWeb [Электронный ресурс]. URL: https://kitware.github.io/paraviewweb/index.html
33. Visualizer [Электронный ресурс]. URL: https://kitware.github.io/visualizer/
34. ParaView [Электронный ресурс]. URL: https://www.paraview.org/
35. Markdown Guide [Электронный ресурс]. URL: https://www.markdownguide.org/
36. The Official YAML Web Site [Электронный ресурс]. URL: https://yaml.org/

# Оглавление