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

Препринты ИПМ • Препринт № 45 за 2025 г.

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

Д.С. Бойков, Н.И. Тарасов,
А.С. Болдарев, К.С. Кондратьев

Интеграция вычислительной платформы MARPLE в цифровую платформу KIAM Digital Tool для решения задач мультифизики

Рекомендуемая форма библиографической ссылки: Интеграция вычислительной платформы MARPLE в цифровую платформу KIAM Digital Tool для решения задач мультифизики / Д.С. Бойков [и др.] // Препринты ИПМ им. М.В.Келдыша. 2025. № 45. 26 с. EDN: ZJETDI
https://library.keldysh.ru/preprint.asp?id=2025-45

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

Д.С. Бойков, Н.И. Тарасов, А.С. Болдарев, К.С. Кондратьев

Интеграция вычислительной платформы MARPLE в цифровую платформу KIAM Digital Tool для решения задач мультифизики

Москва — 2025

Д.С. Бойков, Н.И. Тарасов, А.С. Болдарев, К.С. Кондратьев

Интеграция вычислительной платформы MARPLE в цифровую платформу KIAM Digital Tool для решения задач мультифизики

В работе представлено описание интеграции вычислительной платформы MARPLE в цифровую платформу KIAM Digital Tool для решения сложных мультифизических задач. MARPLE, изначально разработанный как исследовательский код для моделирования высокоскоростных процессов в плазме под воздействием интенсивных энергетических потоков, эволюционировал в универсальный инструмент для численного моделирования задач механики сплошных сред.

MARPLE основан на современных вычислительных технологиях, поддерживающих использование блочно-структурированных и неструктурированных сеток с элементами различной топологии (тетраэдры, гексаэдры, призмы), и реализует методы высокого разрешения для сохранения законов сохранения. Архитектура кода построена на принципах объектно-ориентированного и обобщенного программирования (C++), что обеспечивает гибкость и модульность. Подготовка расчетных областей осуществляется как с помощью внешней CAD/CAE-платформы SALOME, так и через инструменты цифровой платформы KIAM Digital Tool, которые обеспечивают автоматизацию построения сеток, параметризацию геометрии. Параллельные вычисления выполняются в распределенной памяти с использованием MPI.

Цифровая платформа KIAM Digital Tool предоставляет инфраструктуру для автоматизации вычислительных экспериментов, динамического встраивания приложений и управления распределенными ресурсами. Интеграция MARPLE в KIAM Digital Tool позволила реализовать эффективное взаимодействие между моделями физических процессов и вычислительными ресурсами, обеспечивая масштабируемость, повторяемость результатов и удобство управления сложными мультифизическими вычислительными экспериментами.

Приведены примеры решения модельных задач теплопроводности. Предложенный подход открывает перспективы для дальнейшего развития компьютерного моделирования, упрощает интеграцию новых алгоритмов и способствует оптимизации вычислительных экспериментов на суперкомпьютерных системах.

Ключевые слова: мультифизическое моделирование, MARPLE, KIAM Digital Tool, высокопроизводительные вычисления, объектно-ориентированное проектирование, неструктурированные сетки, интеграция программных комплексов, цифровая платформа, вычислительная платформа

D.S. Boykov, N.I. Tarasov, A.S. Boldarev, K.S. Kondratyev

Integration of the Computational Platform MARPLE into the Digital Platform KIAM Digital Tool for Solving Multiphysics Problems

This work presents the integration of the computational platform MARPLE into the digital framework KIAM Digital Tool to address complex multiphysics problems. Originally developed as a research code for modeling high-speed plasma processes under intense energy fluxes, MARPLE has evolved into a versatile tool for numerical simulations in continuum mechanics.

MARPLE leverages modern computational technologies, supporting structured and unstructured meshes with diverse element topologies (tetrahedra, hexahedra, prisms) and employing high-resolution methods to preserve conservation laws. Its architecture adheres to object-oriented and generic programming principles in C++, ensuring flexibility and modularity. Preparation of computational domains is facilitated through both the external CAD/CAE platform SALOME and the tools within KIAM Digital Tool, which automate mesh generation and geometry parametrization. Parallel computations are executed in distributed memory environments using MPI.

The digital platform KIAM Digital Tool provides an infrastructure for automating computational experiments, dynamically embedding applications, and managing distributed resources. Integrating MARPLE into KIAM Digital Tool enables efficient interaction between physical process models and computational resources, enhancing scalability, result reproducibility, and ease of managing complex multiphysics simulations.

Examples of solutions to model heat conduction problems are presented. This approach advances computational modeling, simplifies integration of new algorithms, and optimizes high-performance computing (HPC) experiments on supercomputer systems.

Keywords: multiphysics modeling, MARPLE, KIAM Digital Tool, high-performance computing, object-oriented design, unstructured grids, integration of software complexes, digital platform, computational platform

Оглавление

Введение

Современные задачи численного моделирования в науке и инженерии требуют комплексного подхода, объединяющего точность физических моделей, эффективность вычислительных алгоритмов и масштабируемость на высокопроизводительных системах. Одним из ключевых направлений развития является интеграция специализированных вычислительных платформ в унифицированные цифровые экосистемы, которые обеспечивают автоматизацию процессов, управление ресурсами и взаимодействие между компонентами программного обеспечения. В данной работе рассмотрена интеграция объектно-ориентированного параллельного мультифизического кода MARPLE [1] в цифровую платформу KIAM Digital Tool (KDT) [2], разработанную в Институте прикладной математики им. М.В. Келдыша РАН.

MARPLE изначально создавался как исследовательский код для моделирования высокоскоростных процессов в плазме под воздействием интенсивных энергетических потоков. Его аббревиатура происходит от «Magnetically Accelerated Radiative Plasma Explorer», что отражает первоначальную цель — анализ экспериментов с высокоплотной импульсной плазмой. За годы развития MARPLE превратился в универсальную вычислительную платформу, способную решать задачи механики сплошных сред в различных предметных областях: физике плазмы, астрофизике, термомеханике твёрдого тела и других. Основными преимуществами платформы являются использование неструктурированных и блочно-структурированных сеток с элементами различной топологии (тетраэдры, гексаэдры, призмы), реализация методов высокого разрешения для сохранения законов сохранения, а также объектно-ориентированная архитектура, позволяющая гибко адаптировать код под новые задачи.

Однако эффективное использование таких платформ требует интеграции с цифровыми инфраструктурами, обеспечивающими автоматизацию вычислительных экспериментов, управление распределёнными ресурсами и взаимодействие с внешними приложениями. Цифровая платформа KDT представляет собой такую систему, которая объединяет возможности суперкомпьютерных вычислений, веб-интерфейсов и динамического встраивания программного обеспечения. Она позволяет автоматизировать подготовку расчётных областей, параметризацию геометрии, запуск задач на удалённых вычислительных ресурсах и анализ результатов, обеспечивая низкие системные требования для пользователей и поддержку многопользовательского режима [3].

Интеграция MARPLE в KDT открывает новые возможности для решения мультифизических задач, включая моделирование обтекания объектов потоками газа, тепломассоперенос в сложных средах и взаимодействие физических процессов в условиях экстремальных энергетических нагрузок. Это позволяет не только повысить масштабируемость и повторяемость вычислений, но и упростить внедрение новых алгоритмов и физических моделей. Примеры

применения интегрированной системы к задачам динамики плазмы, астрофизики и прочностного анализа демонстрируют ее универсальность и эффективность.

Дальнейшая структура работы включает описание архитектуры MARPLE и его ключевых физических моделей, обзор возможностей KDT, детали интеграции и результаты тестовых расчётов. Предложенный подход направлен на развитие моделирования и оптимизацию использования суперкомпьютерных ресурсов в научных и промышленных исследованиях.

Цифровая платформа KIAM Digital Tool

Одной из ключевых особенностей KDT является возможность взаимодействия с удалёнными суперкомпьютерами через веб-интерфейс, что значительно упрощает процесс выполнения сложных вычислений и обработки данных.

Платформа взаимодействует с удалёнными вычислительными ресурсами посредством SSH-протокола, что позволяет осуществлять запуск пользовательских заданий на удалённых вычислителях, передавать файлы и управлять задачами без необходимости использования командной строки, средствами графического веб-интерфейса. При этом доступ пользователя строго контролируется и может быть ограничен в целях информационной безопасности.

KDT предоставляет широкий спектр функциональных возможностей, направленных на обеспечение удобного и продуктивного взаимодействия с удалёнными вычислительными ресурсами, включая суперкомпьютерные кластеры. Чтобы провести подготовку к вычислительному эксперименту, пользователю предоставляется возможность выбрать вычислительный ресурс и использовать интерфейс для введения исходных данных, сгенерированный на основе паспорта приложения.

Не менее важной является функциональность, связанная с запуском и управлением вычислительными задачами. Платформа позволяет инициировать процессы на удалённых серверах и в то же время отслеживать их выполнение в режиме реального времени. Пользователь получает информацию о текущем статусе задачи, прогрессе её выполнения и результатах по завершении. Такой механизм оказывается особенно востребованным при решении ресурсоёмких задач, выполнение которых может занимать значительное время и требовать тщательного контроля.

Значительное внимание в платформе уделено визуализации данных. Для пользователей доступны инструменты анализа и отображения результатов моделирования в виде графиков, диаграмм и трёхмерных представлений. Это особенно важно для научных и инженерных задач, где не только численные значения, но и наглядность представления информации играют решающую роль в интерпретации и принятии решений.

Архитектурное исполнение и взаимодействие компонентов

Разработанная цифровая платформа реализована в виде клиент-серверной системы, поддерживающей динамическую интеграцию удалённых вычислительных ресурсов и проблемно-ориентированных приложений для проведения расчётов на суперкомпьютерных комплексах. Серверная часть системы построена с использованием фреймворка NestJS [4], обеспечивающего маршрутизацию, управление сеансами и взаимодействие с реляционной базой данных, и функционирует в среде Node.js [5]. Архитектура взаимодействия сервера веб-лаборатории представлена на рис. 1.

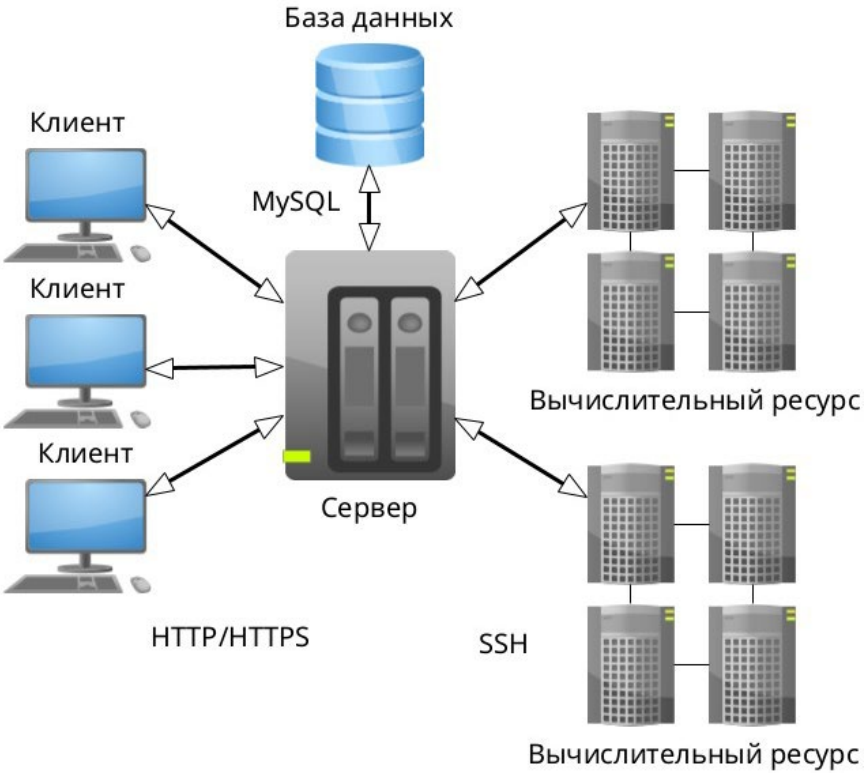

Рис. 1. Схема взаимодействий цифровой платформы

Клиентская часть системы реализована как одностраничное веб-приложение (SPA) с использованием реактивного фреймворка Vue.js [6]. При инициализации сеанса через веб-браузер пользователю предоставляются HTML/CSS-макеты и JavaScript-скрипты для отображения интерфейса. Дополнительные данные из базы извлекаются через постоянное веб-сокет-соединение, устанавливаемое после прохождения процедуры аутентификации.

Организация доступа и управления ресурсами

Для обеспечения контролируемого доступа к системе реализован механизм регистрации с использованием уникального ключа, выделяемого администратором. На основе ключа формируются роли пользователей:

- Расчётчик — выполняет вычислительные эксперименты;

- Разработчик ПО — интегрирует приложения в систему;
- Администратор — управляет базой данных и регистрирует вычислительные ресурсы.

После регистрации в системе создаются записи о связях пользователя с вычислительными ресурсами, включая рабочие директории, имена пользователей и генерируемый SSH-RSA-ключ для авторизации на удалённых узлах. Взаимодействие с ресурсами осуществляется через протокол SSH с передачей команд и данных файловой системы.

Интеграция приложений

Интеграция проблемно-ориентированных приложений реализуется через создание сущности «Приложение», описываемой следующими атрибутами:

- Название (отображается в интерфейсе);
- Идентификатор (используется в сценариях);
- Список разработчиков.

Исходный код хранится в репозитории Git и инициализируется одним из способов:

- Пустой каталог;
- Архив, загружаемый пользователем;
- Клонирование существующего репозитория.

Подготовка приложений на удалённых ресурсах

Перед использованием приложения необходимо подготовить его на удалённых вычислительных ресурсах, доступных пользователю. Для этого на вкладке «Подготовка» страницы приложения (рис. 2) выбирается версия (git-ветвь) для целевого ресурса и активируется кнопка «Развернуть».

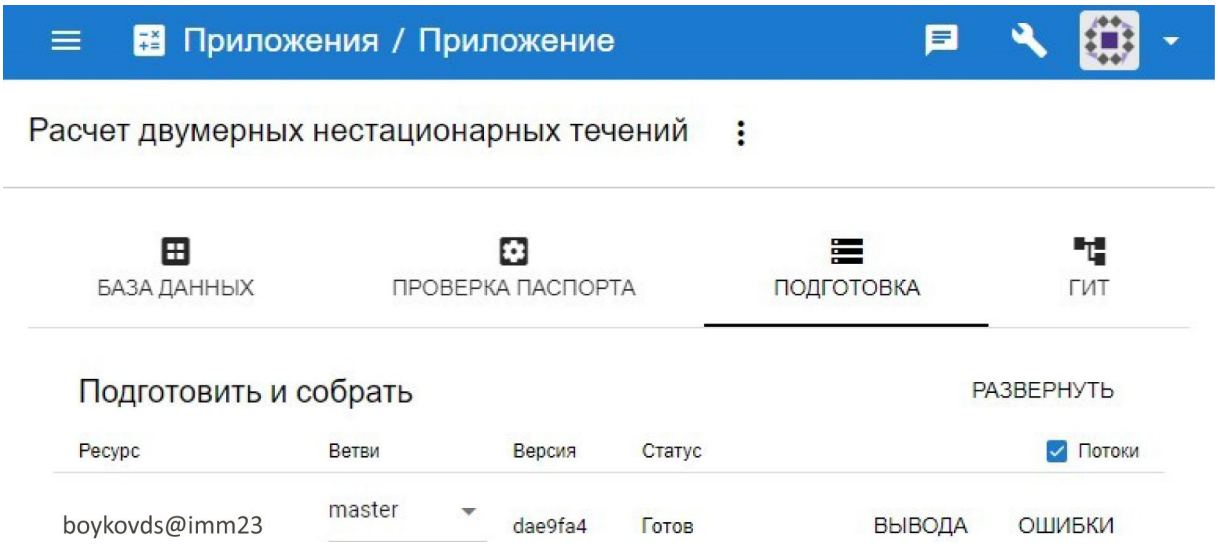

Рис. 2. Интерфейс подготовки приложения

Сервер веб-лаборатории передаёт исходный код указанной версии на удалённый узел в рабочую директорию и выполняет команду сборки, определённую в паспорте приложения. Статус операции отображается в реальном времени через веб-интерфейс.

Автоматизация вычислений

Для автоматизации процессов (сборка, запуск, конфигурация) приложение должно содержать YAML-файл «passport.yaml», описывающий:

- Команду сборки;
- Дерево входных параметров;
- Список выходных данных;
- Конфигурационные зависимости.

На основе данных из паспорта система обеспечивает:

- Компиляцию кода на удалённом ресурсе;
- Генерацию графического интерфейса для ввода параметров;
- Мониторинг выполнения;
- Доступ к результатам вычислений.

Пример структуры файла «passport.yaml» приведён на рис. 3.

Для группировки приложений в функциональные блоки используется сущность «Сценарий». Основные характеристики:

- Связь приложений через входные/выходные параметры;
- Определение последовательности запуска.

Создание сценария включает:

1. Задание названия и списка разработчиков;
2. Инициализацию репозитория;
3. Формирование файла «script.yaml» с описанием:
   - Используемых приложений (по идентификаторам);
   - Межкомпонентных связей;
   - Этапов исполнения.

Формализованное описание сценариев позволяет системе генерировать унифицированный интерфейс для вычислительных экспериментов. Пример метаописания приведён на рисунке 3.

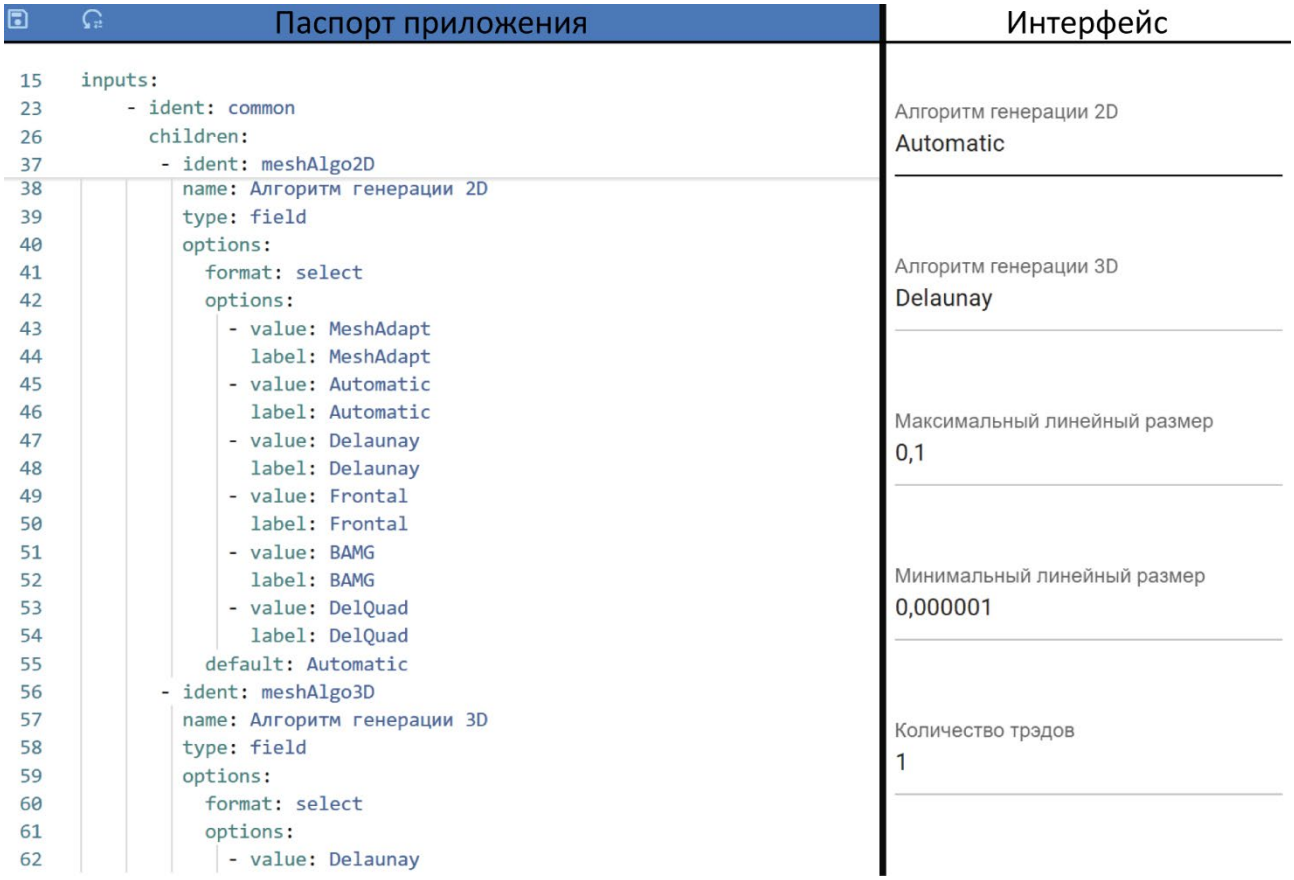

Рис. 3. Фрагмент паспорта приложения (слева) и пример сгенерированного на его основе веб-интерфейса (справа)

Вычислительная платформа MARPLE

В настоящей работе представлен объектно-ориентированный, параллельный многомерный эйлеров мультифизический программный комплекс MARPLE, разработанный в Институте прикладной математики им. М.В. Келдыша Российской академии наук для проведения научного численного моделирования с использованием высокопроизводительных вычислений [7, 8].

Первоначально код MARPLE [9] создавался как исследовательский инструмент для моделирования процессов, возникающих в экспериментах с высокоплотной импульсной плазмой. В ходе дальнейшего развития он был адаптирован и стал универсальным программным средством, применимым к широкому классу задач из различных предметных областей.

Код реализует множество структур данных и алгоритмов, независимых от специфики решаемых задач. Особое внимание уделено обработке неструктурированных сеток — сложной задаче, особенно при распределённом исполнении. В MARPLE разработаны эффективные структуры данных для представления и обработки неструктурированных сеток, что позволяет обеспечить гибкость и масштабируемость вычислений. Таким образом, на данный момент MARPLE представляет собой многофункциональную

вычислительную платформу, предназначенную для выполнения многоцелевых численных экспериментов.

Программа применялась для решения задач из следующих областей:

- физика плазмы высокой плотности энергии, пинчи;
- капиллярный разряд;
- астрофизика (моделирование эволюции нейтронных звёзд);
- механика твёрдого тела, анализ воздействия сильных энергетических потоков на конденсированное вещество.

MARPLE поддерживает моделирование трёхмерных процессов в областях сложной геометрической формы на нерегулярных сетках, включая блочные сетки, состоящие из элементов различного типа. Допускается использование трёхмерных элементов, топологически эквивалентных параллелепипедам, тетраэдрам, треугольным призмам и четырёхгранным пирамидам; грани элементов могут быть неплоскими. Для двумерных моделей поддерживаются треугольные и четырёхугольные элементы.

Реализованные физические модели

В состав программного комплекса включены следующие основные физические модели:

- одножидкостная двухтемпературная магнитогидродинамическая модель динамики плазмы;
- модель диффузии электромагнитного поля;
- модель теплового баланса плазмы;
- двухжидкостная магнитогидродинамическая модель;
- модель течения многокомпонентной плазмы;
- модель деформируемого твёрдого тела;
- материальные уравнения.

Численное решение задач осуществляется с использованием широкодиапазонных уравнений состояния, а также моделей переноса и оптических свойств веществ. MARPLE поддерживает работу с блочно-структурированными и неструктурированными сетками, включающими тетраэдрические, гексаэдрические, призматические и другие элементы. Алгоритмы солверов основаны на применении методов высокого разрешения, обеспечивающих точное соблюдение законов сохранения. Для интегрирования полной системы уравнений используется подход физического расщепления [10].

Архитектура программного обеспечения

Архитектура MARPLE построена на принципах объектно-ориентированного проектирования и обобщённого программирования, реализованного на языке C++. Код разделён на модули: аппроксимация, работа с сетками, солверы, граничные условия и уравнения состояния. Иерархия

классов в модулях, а также связи между ними представлены на рис. 4 (показаны только необходимые классы и связи, модули выделены рамками).

Основные модули системы:

- MARPLE_Driver — главный модуль, управляющий вычислительным процессом, координирующий взаимодействие между другими модулями.
- Main_Solver — модуль верхнего уровня, осуществляющий операции с физическими субдоменами.
- PhysicalDomainSolver — физический солвер, выполняющий операции на уровне физических подобластей и управляемый MainSolver. Каждый объект PhysicalDomainSolver выполняет операции на одном и только одном вычислительном узле и в одной и только одной физической подобласти. Физическая подобласть, соответственно, назначается конкретному объекту PhysicalDomainSolver.
- ElementalSolver — солверы выполняют вычисления для решения определенной подсистемы уравнений или только одного уравнения в соответствующей физической подобласти. Многие объекты ElementalSolver обеспечивают аппроксимацию полной системы уравнений, которую необходимо решить в соответствующей физической подобласти, на основе принципа суммарной аппроксимации. Например, если необходимо решить уравнения теплопроводности, джоулева нагрева и переноса излучения в конкретной физической подобласти, то один элементарный солвер может использоваться для численного решения уравнения теплопроводности, а другой может решать оставшиеся два. В этом примере два элементарных солвера обеспечивают суммарную аппроксимацию всех трёх уравнений, определённых в конкретной физической подобласти. Каждый объект ElementalSolver выполняет операции над одним и только одним вычислительным узлом, одной и только одной физической подобластью и, кроме того, решает одно конкретное уравнение или подмножество системы уравнений. Элементарные солверы полностью контролируются солверами физических подобластей PhysicalDomainSolver, описанных выше.
- MatterProperties — модуль, обеспечивающий унифицированый доступ к уравнениям состояния для каждой физической подобласти.

Рис. 4. Диаграмма классов UML: архитектура MARPLE

Параллельные вычисления

Решение мультифизических задач требует обработки значительного количества переменных в каждом узле сетки на каждом временном шаге. Для моделирования реальных трёхмерных устройств и многомасштабных процессов необходимы сетки, содержащие миллионы и даже сотни миллионов ячеек. Такие задачи могут быть выполнены только с использованием распределённых вычислений. MARPLE поддерживает запуск на системах с общей или распределённой памятью в режиме MPI, используя декомпозицию области и сетки. Для декомпозиции сетки используются библиотеки METIS/ParMETIS [11], обеспечивающие эффективное распределение нагрузки между процессорами. Проектная система сборки реализована с использованием CMake [12], что позволяет использовать различные среды разработки и компиляторы, сохраняя кроссплатформенность.

Код оптимизирован для работы в распределённой памяти (MPI) на современных высокопроизводительных вычислительных системах.

Использование стороннего программного обеспечения

Для подготовки расчётных сеток используется открытая CAD/CAE-платформа SALOME [13], предоставляющая средства описания геометрии, задания граничных условий, генерации и адаптации сеток. Визуализация результатов моделирования производится с помощью ParaView [14].

Для решения больших разреженных систем линейных уравнений, возникающих при неявных аппроксимациях, применяется библиотека Aztec [15, 16], поддерживающая параллельные вычисления и совместимая с GPGPU. Aztec использует подпрограммы из библиотек LAPACK и BLAS.

Интеграция свободного программного обеспечения является важной частью технологии разработки MARPLE. Это способствует повышению надёжности, переносимости, упрощает отладку и взаимодействие с внешними кодами.

Интеграция MARPLE & KIAM Digital Tool

Ключевой проблемой интеграции платформы MARPLE и KDT стала несовместимость форматов конфигурационных файлов.

Вычислительная платформа MARPLE изначально не предусматривала поддержку таких форматов данных, как JSON, используя вместо этого специализированный формат конфигурационного файла в текстовом формате. Чаще всего в контексте использования MARPLE этот файл назывался usertask, поэтому в дальнейшем будем говорить о первоначальном формате конфигурационного файла MARPLE как о формате usertask. Причина использования своего нестандартного формата была в желании сохранить для конфигурационного файла простой и интуитивно понятный формат типа <величина> = <значение>, без необходимости для пользователя при ручном

редактировании конфигурационных параметров соблюдать формальные синтаксические правила (например, закрывать всякий открытый тег, как в XML). Однако при интеграции кода в оболочку KDT возникла ситуация, когда файл генерируется автоматически интерактивно-графической оболочкой. В KDT для конфигурационных файлов принят формат JSON. В результате возникла необходимость разработки программного решения для преобразования одного формата данных (JSON) в формат, совместимый с MARPLE.

Изначально рассматривались несколько возможных вариантов решения данной задачи. Все эти подходы предполагали изменить существующий исходный код MARPLE. Однако этот вариант требовал значительных усилий от разработчиков и траты большого количества времени. Учитывая эти факторы, было принято решение о разработке отдельной утилиты, выполняющей функцию преобразования данных и интегрируемой в цифровую платформу KDT.

Также большую роль сыграло формирование паспорта приложения, по его полям строятся универсальный пользовательский интерфейс и конфигурационный JSON файл.

Описание разрабатываемой утилиты

Утилита создана в рамках решения проблемы взаимосвязи платформ KDT и MARPLE. Основное назначение созданного кода — конвертация передаваемых данных из JSON в USERTASK. На рисунке 5 представлена диаграмма, демонстрирующая взаимодействие KDT и MARPLE до разработки утилиты.

Рис. 5. Схема взаимодействия KDT и MARPLE без JsonProcessor

Для реализации утилиты была выбрана библиотека nlohmann/json [17], которая соответствует требованиям, предъявляемым к JSON-библиотеке в рамках данного проекта. Она реализована в виде чисто заголовочного модуля, не требует дополнительных зависимостей и сборки, а также использует современные средства C++, соответствуя стандартам, начиная с C++11.

Проектирование JsonProcessor осуществлялось с учётом принципов объектно-ориентированного программирования и рекомендаций SOLID, что позволило создать модульную и расширяемую систему с высокой степенью абстракции и слабой связностью компонентов. Центральным звеном архитектуры выступает класс JsonProcessor, реализующий поведенческий паттерн Фасад. Его задача — координировать работу подсистем, предоставляя единый интерфейс для запуска процесса обработки данных. Такой подход позволяет скрыть сложность внутренней реализации, обеспечивая простоту использования и изоляцию зависимостей между модулями (см. рис. 6).

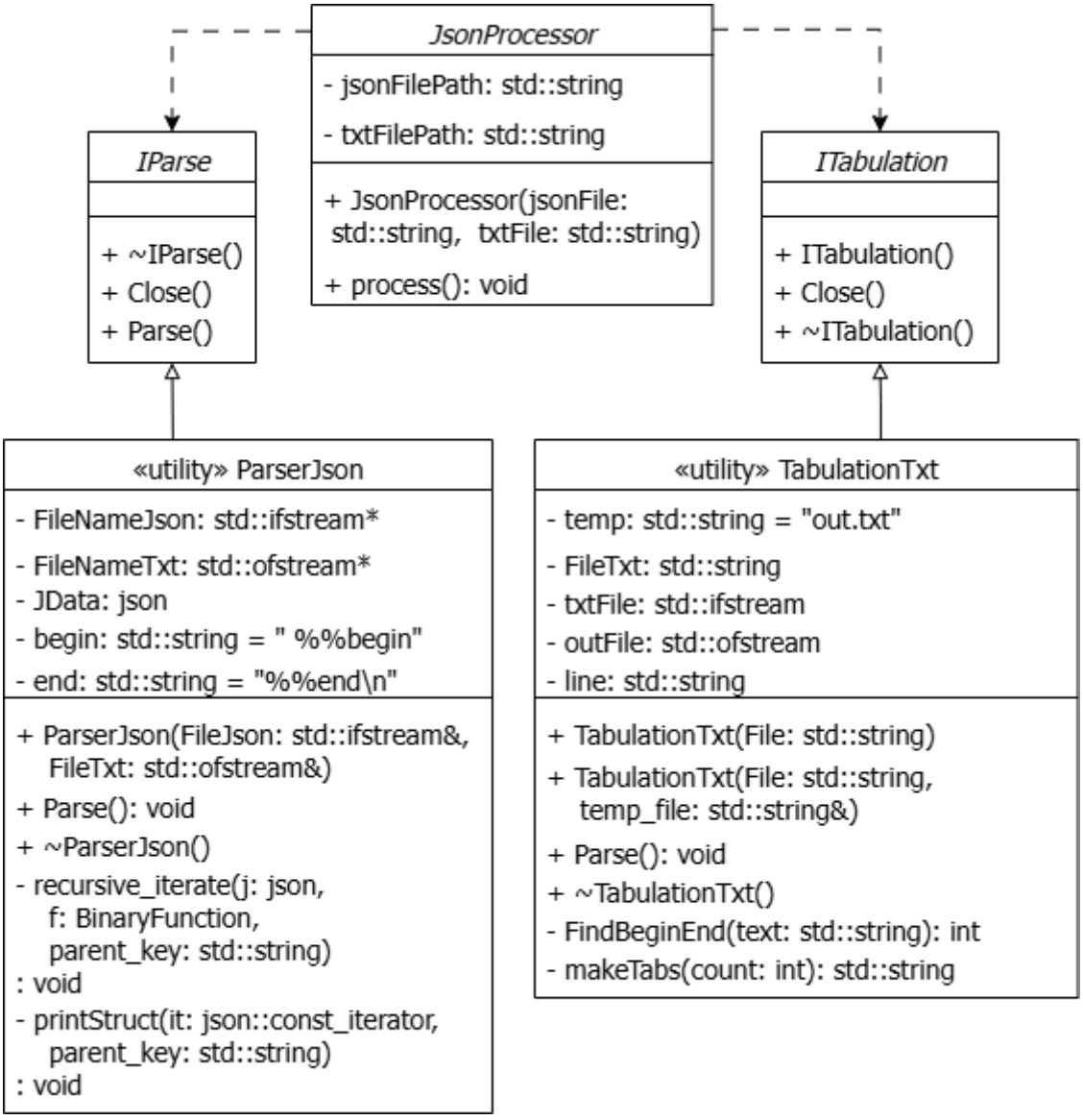

Рис. 6. UML диаграмма классов утилиты JsonProcessor

Фасад управляет последовательным выполнением двух этапов: сначала с помощью класса ParserJson производится парсинг входного JSON-файла, а затем результат передаётся в класс TabulationTxt, который форматирует полученные данные, добавляя табуляцию для повышения читаемости. ParserJson, в свою очередь, взаимодействует с библиотекой nlohmann::json, обеспечивая высокоуровневый и универсальный доступ к структуре JSON-объектов. Это позволяет обрабатывать произвольные входные данные, не завися от их конкретных ключей или структуры. Ключевая функция этого класса — десериализация JSON и подготовка данных к сохранению в формате USERTASK.

Сформированный файл с данными, которые хранят начальные значения для вычислительного эксперимента, передаются между KDT и MARPLE через код утилиты JsonProcessor, которая осуществляет целостность и сохранность данных в удобном формате для используемых платформ (см. рис. 7).

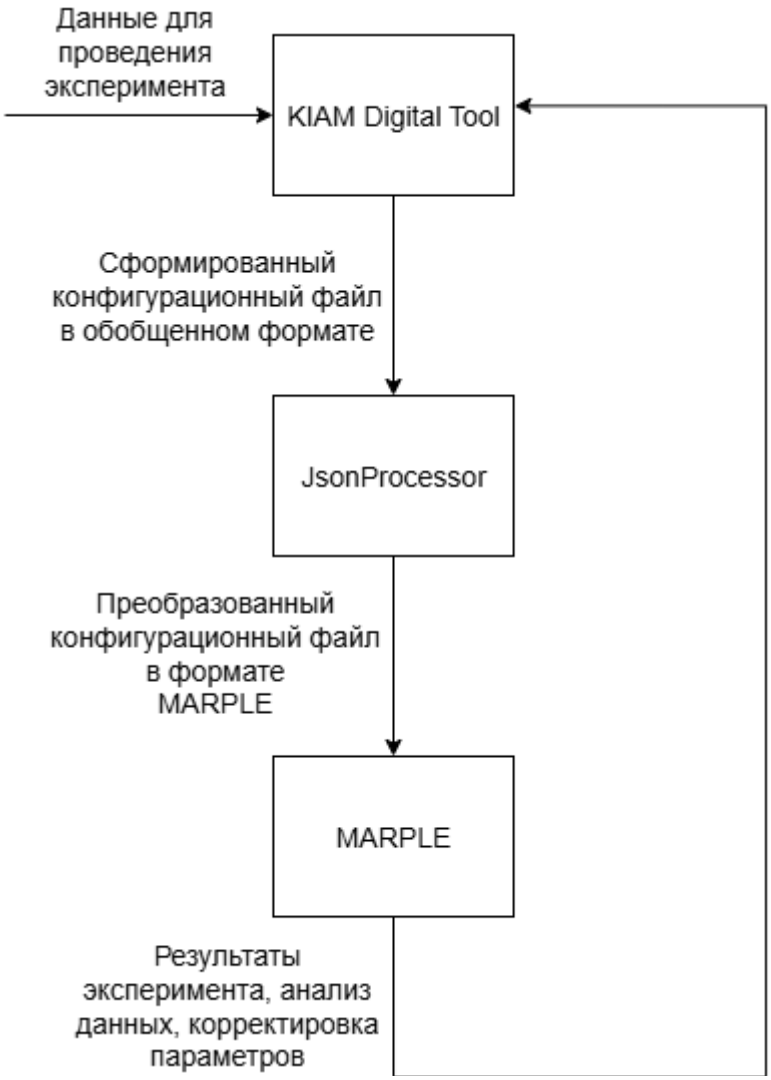

Рис. 7. Схема взаимодействия систем KDT и MARPLE

Особую роль в архитектуре утилиты играют интерфейсы IParse и ITabulation, реализованные соответственно в классах ParserJson и TabulationTxt. Их внедрение позволило реализовать принцип инверсии зависимостей (DIP), сделав фасад JsonProcessor независимым от конкретных реализаций парсинга и форматирования. Вместо работы с жёстко заданными классами JsonProcessor взаимодействует с абстракциями, что делает систему гибкой и открытой к расширению без изменения существующего кода. Такой подход также соответствует принципу разделения интерфейсов (ISP) — каждый интерфейс определяет строго необходимый набор операций, — а также принципу открытости/закрытости (OCP), позволяя добавлять новый функционал без вмешательства в уже проверенный и протестированный код.

Вычислительные эксперименты

Вычислительный эксперимент для поставленной задачи основывается на использовании группы проблемно-ориентированных приложений, связи между которыми изображены на рисунке 8.

Они предназначены:

mesh (meshgen) — для генерации сетки в формате gmesh;

conv — преобразует сетки из формата gmesh в формат marple;

confGen — преобразует входящий json в usertask;

marple — MARPLE, используемый для решения поставленной задачи.

Рис. 8. Схема взаимодействия приложений

Настройка осуществляется на этапе «Входные параметры», где на основании сценария и паспортов приложений генерируется дерево форм, изображённое на рисунке 9.

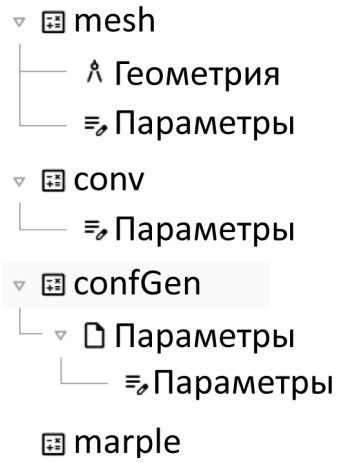

Рис. 9. Дерево форм

При помощи данного дерева пользователь задаёт начальные и исходные параметры конфигурации и геометрические модели для каждого этапа перед запуском расчёта.

Постановка задачи

Для демонстрации работы системы рассмотрим моделирование процесса теплопереноса в твёрдом прямоугольном стержне. Основой модели в этом случае выступит уравнение теплопроводности в следующем виде:

$$\frac{\partial T}{\partial t} = \frac{\partial}{\partial s} \cdot \left(k_0 \cdot T^{\alpha} \cdot \frac{\partial T}{\partial s} \right), \tag{1}$$

где T — искомая функция (температуры), κ_0, a — свободные коэффициенты, $s \equiv x|y|z$. Это уравнение дополняется следующими начальными и граничными условиями:

$$T(s,0) = \begin{cases} \left[\frac{\alpha D}{k_0}\left(s_0 - s\right)\right]^{\frac{1}{\alpha}}, & s \le s_0, \\ 0, & s > s_0. \end{cases} \qquad (2)$$

$$T(0,t) = \left[\frac{\alpha D}{k_0}\left(Dt + s_0\right)\right]^{\frac{1}{\alpha}}, \quad t > 0,$$

где D — скорость распространения волны, s_0 — свободный параметр.

Представленные уравнения дискретизируются по пространственным переменным проекционно-сеточным методом на основе метода конечных объёмов. В аппроксимации производной по времени используется схема Кранка-Николсон. Для аппроксимации пространственных операторов второго порядка используются схемы на основе метода Галёркина с разрывными базисными функциями. В практических расчётах шаг по времени, как правило, определяется требованиями физической точности и допускает использование схем первого порядка для интегрирования по времени без снижения общей точности решения. Предлагаемая методика, с одной стороны, позволяет решать задачу теплопроводности на неструктурированных трёхмерных сетках, а использование явно-неявной временной методики, с другой стороны, существенно ускоряет время расчёта.

Тест. Бегущая волна

Задача бегущей волны в теплопроводности — это частный тип задачи теплопроводности, в которой ищется решение в виде бегущей волны. Это означает, что температура в теле не меняется произвольно во времени и пространстве, а распространяется с постоянной формой и скоростью, как волна.

Для теста использовались следующие значения свободных коэффициентов: a = 2.0, κ = 0.5, s_0 = 0.5, D = 5.0; модельное время изменялось от 0 до 0.4 мкс с постоянным шагом $\tau = 2 \cdot 10^{-4}$ [18]. В качестве области исследования рассматривался стержень с размерными параметрами: 1x1x4 см. Расчётная сетка содержит 2e+4 тетраэдров и построена с использованием KDT.

Результаты моделирования

Представленные приложения подготовлены на удалённом вычислителе на основании представленных метаописаний. Кроме того, их взаимосвязи описываются сценарием — специальным текстовым файлом в формате YAML, на основании которого формируется веб-страница конкретного расчёта.

Для проведения вычислительного эксперимента перейдём в проект (проекты позволяют каталогизировать несколько расчётов) «Marple Playground» в веб-лаборатории KIAM. В его рамках создадим «Расчёт» с названием «Бегущая волна».

На первом этапе вычислительного эксперимента, состоящего из 4 этапов (mesh, conv, confGen и marple), выберем вычислительные ресурсы для используемых программ посредством графического интерфейса, изображённого на рисунке 10.

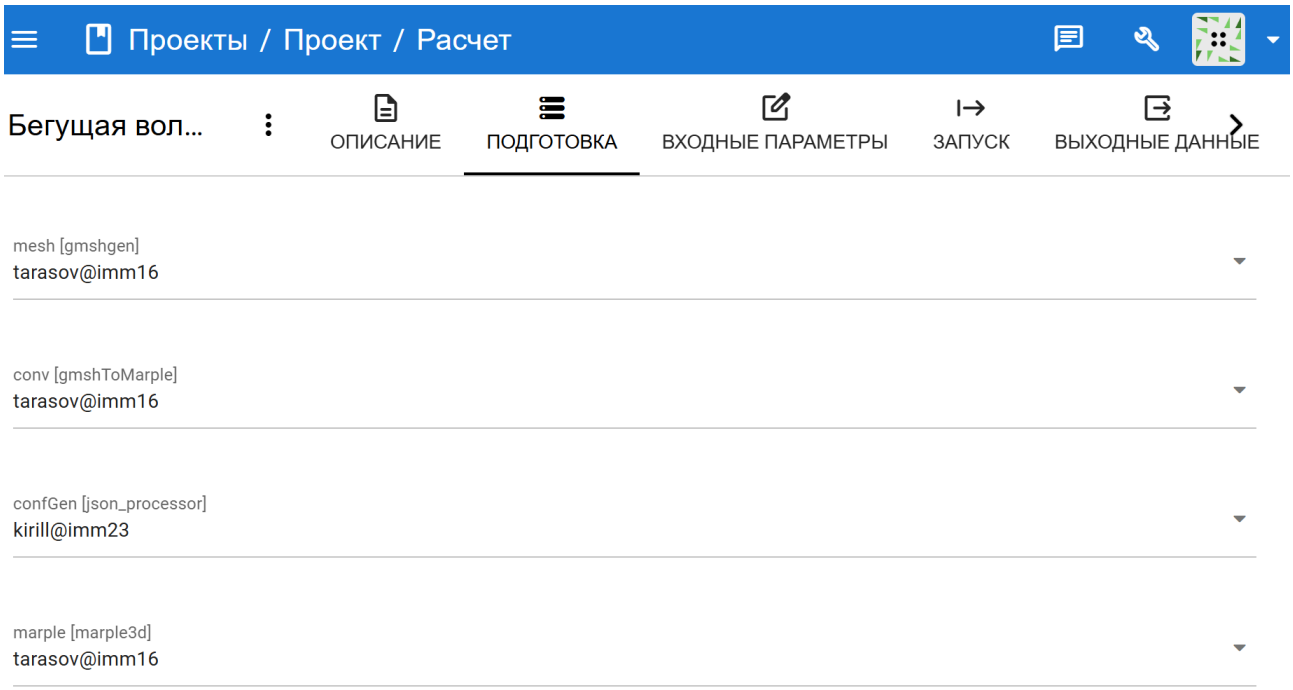

Рис. 10. Выбор вычислительных ресурсов

После выбора вычислительных ресурсов система подготовит рабочие директории приложений, а также установит их актуальные версии, в результате чего становится возможным осуществить настройку и удалённый запуск данных программ.

После того как начальные данные были указаны на вкладке «Запуск», можно запустить все этапы и посмотреть вывод и логи каждого этапа, включая сообщения об ошибке. Его графический интерфейс представлен на рисунке 11.

Параметры

```
{
    "HeatConductionSolver": {
        "nabla_version": 1,
        "two_temp_mode": 2,
        "XMAX": "HC_Const",
        "XMIN": "HC_Test",
        "YMAX": "HC_Sym",
        "YMIN": "HC_Sym",
        "ZMAX": "HC_Sym",
        "ZMIN": "HC_Sym"
    },
    "HC_Const": {
        "value": 0
    },
    "IdealGas": {
        "rho_vacuum": 1e-6,
        "T_vacuum": 0.001,
        "ignore_vacuum_in_timestep_calc_K": 1,
        "vacuum_flow_limitation_K_l": 1,
```

Рис. 11. Подготовка начальных параметров confGen

На рисунке 12 демонстрируется полученный конфигурационный файл после обработки этапа confGen (JsonProcessor).

Обработанный TXT

```
HC_Const = %%begin
    value = 0
%%end
HeatConductionSolver = %%begin
    XMAX = HC_Const
    XMIN = HC_Test
    YMAX = HC_Sym
    YMIN = HC_Sym
    ZMAX = HC_Sym
    ZMIN = HC_Sym
    nabla_version = 1
    two_temp_mode = 2
%%end
IdealGas = %%begin
    Atomic_mass = 1
    Atomic_number = 1
    GAS_GAMMA = 1.6666666666666667
    T_vacuum = 0.001
    ignore_vacuum_in_timestep_calc_K = 1
    rho_vacuum = 1e-06
```

Рис. 12. Обработанный JSON из confGen

Перейдём во вкладку «выходные данные», чтобы посмотреть результаты расчёта вычислительного эксперимента.

В результате проверки работоспособности всей системы констатируем, вычислительный эксперимент успешно воспроизведён (см. рис. 13, 14), что

говорит о правильной изначальной конфигурации MARPLE в KDT, таким образом была подтверждена корректная работа разработанной утилиты.

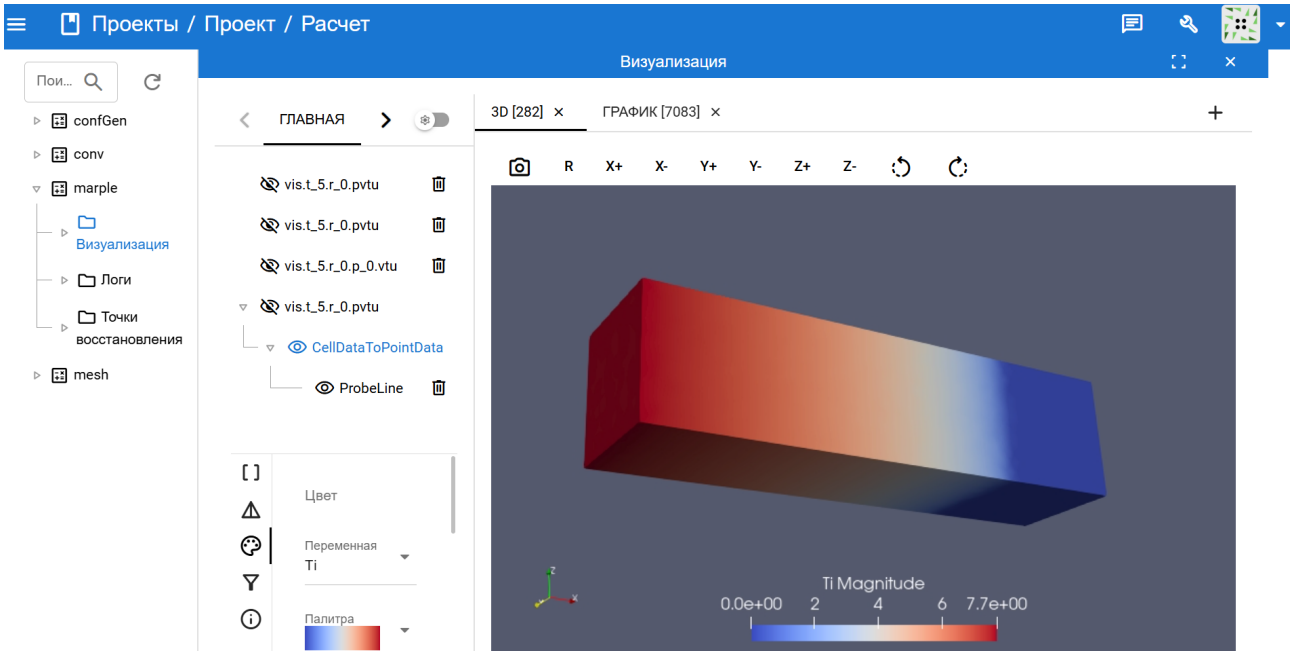

Рис. 13. Визуализация результатов в KDT. Распределение температуры [кэВ] в исследуемом образце

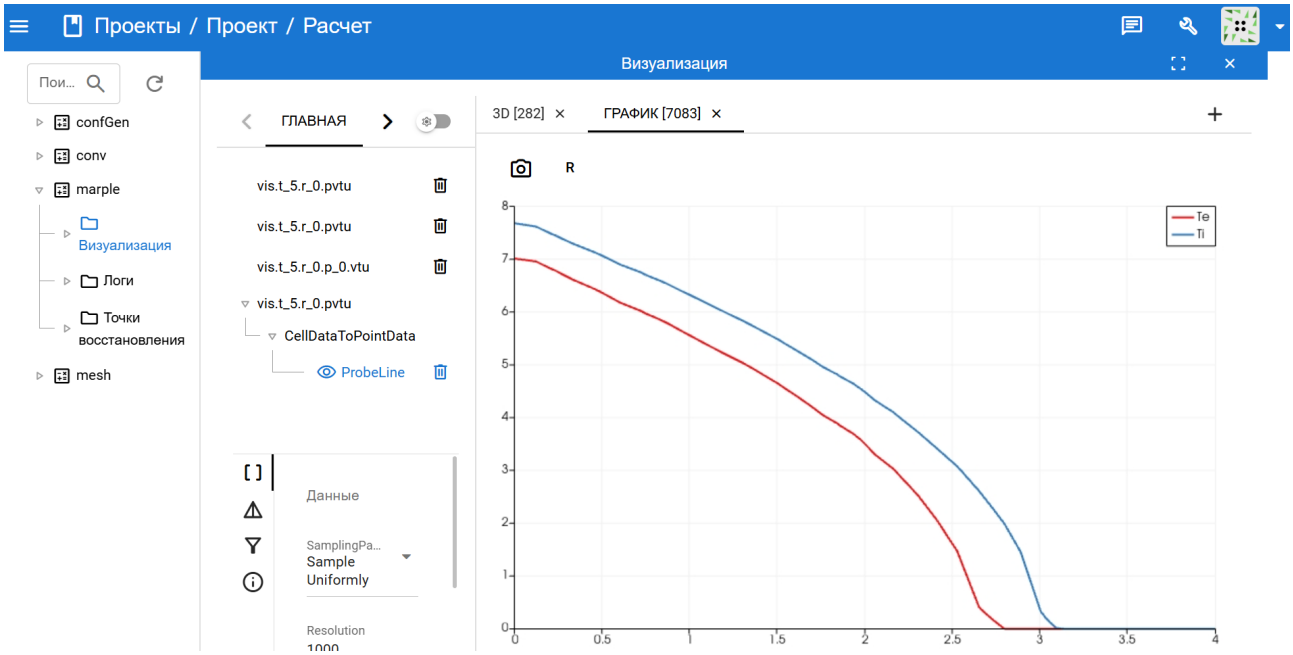

Рис. 14. Визуализация результатов в KDT. График распределения температуры [кэВ] по длине образца

Выводы

В ходе исследования была реализована утилита для конвертации конфигурационного файла MARPLE, которая позволила произвести интеграцию вычислительной платформы MARPLE в цифровую экосистему KIAM Digital Tool, что позволило создать унифицированную систему для решения сложных мультифизических задач. Интеграция MARPLE, изначально разработанного как исследовательский код для моделирования высокоскоростных процессов в плазме, в KIAM Digital Tool обеспечила автоматизацию ключевых этапов вычислительного эксперимента: подготовки расчётных областей, параметризации геометрии, запуска задач на удалённых суперкомпьютерах и анализа результатов.

Основные достижения интеграции включают:

1. Унификация интерфейсов взаимодействия:
   - Обеспечение доступа MARPLE к различным вычислительным ресурсам (суперкомпьютеры, кластеры, облачные системы);
   - Автоматизация передачи данных между графическим интерфейсом KIAM Digital Tool и ядром MARPLE, включая сеточные структуры, физические параметры и уравнения состояния.
2. Модульность и расширяемость:
   - Использование объектно-ориентированной архитектуры MARPLE для интеграции новых физических моделей (например, математических моделей сквозного моделирования) без изменения базовой логики платформы;
   - Поддержка создания неструктурированных и блочно-структурированных сеток с элементами различной топологии (тетраэдры, гексаэдры, призмы) через инструменты KIAM Digital Tool.
3. Повышение эффективности вычислений:
   - Параллелизация задач с использованием MPI и GPU-ускорителей, управляемая через инфраструктуру KIAM Digital Tool;
   - Оптимизация распределения нагрузки на основе декомпозиции сеток с помощью библиотек METIS/ParMETIS.
4. Интерактивная визуализация и анализ:
   - Интеграция ParaView и собственных инструментов KIAM Digital Tool для визуализации многомерных данных, включая поля давления, температуры, магнитных полей;
   - Поддержка анализа результатов в реальном времени с возможностью корректировки параметров модели.

Интеграция MARPLE в KIAM Digital Tool продемонстрировала свою эффективность при решении тестовых задач. Ключевыми преимуществами интеграции стали:

- Снижение порога входа для пользователей за счёт веб-интерфейса, исключающего необходимость локальной установки специализированного ПО;
- Масштабируемость вычислений, позволяющая обрабатывать сетки с миллионами и сотнями миллионов ячеек;
- Междисциплинарная совместимость, обеспечивающая взаимодействие MARPLE с другими модулями KIAM Digital Tool (например, для задания начальных условий нагрузки в результате воздействия ионизирующего излучения).

В дальнейшем планируется расширить функционал платформы за счёт:

- Поддержки адаптивных сеток и многомасштабного моделирования;
- Интеграции методов машинного обучения для прогнозирования поведения физических систем;
- Улучшения безопасности при работе с конфиденциальными данными и оптимизации коммуникации между компонентами.

Результаты работы подтверждают, что интеграция MARPLE в KIAM Digital Tool открывает новые возможности для комплексного моделирования мультифизических процессов, упрощает внедрение инновационных алгоритмов и способствует эффективному использованию суперкомпьютерных ресурсов в научных и промышленных задачах.

Библиографический список

1. Гасилов В.А., Болдарев А.С., Ольховская О.Г., Бойков Д.С., Шарова Ю.С., Савенко Н.О., Котельников А.М. MARPLE: программное обеспечение для мультифизического моделирования в задачах сплошных сред // Препринты ИПМ им. М.В. Келдыша. 2023. № 37. 41 с. https://doi.org/10.20948/prepr-2023-37 https://library.keldysh.ru/preprint.asp?id=2023-37
2. Тарасов Н.И., Подрыга В.О., Поляков С.В. Веб-лаборатория для суперкомпьютерного многомасштабного моделирования задач напыления // Вычислительные методы и программирование. 2023. 24, № 4. 463–484. doi 10.26089/NumMet.v24r432.
3. Лобанова Е.Е., Тарасов Н.И. Применение цифровой платформы для моделирования задач газовой динамики // Препринты ИПМ им. М.В. Келдыша. 2023. № 77. 20 с. https://doi.org/10.20948/prepr-2023-77 https://library.keldysh.ru/preprint.asp?id=2023-77
4. NestJS — A progressive Node.js framework. URL: https://nestjs.com/
5. Node.js. URL: https://nodejs.org/
6. Vue.js — The Progressive JavaScript Framework Vue.js. URL: https://vuejs.org/
7. Gasilov V.A., D'yachenko S.V., Olkhovskaya O.G., Boldarev A.S., Kartasheva E.L., Boldyrev S.N. Object-oriented programming and parallel computing in radiative magnetohydrodynamics simulations. In Parallel

Computing: Architectures, Algorithms and Applications, ParCo 2007, Forschungszentrum Jülich and RWTH Aachen University, Germany, 4–7 September 2007, p. 475–482, 2007.

8. Гасилов В.А., Болдарев А.С., Дьяченко С.В., Ольховская О.Г., Карташева Е.Л., Болдырев С.Н., Багдасаров Г.А., Гасилова И.В., Бояров М.С., Шмыров В.А. Пакет прикладных программ MARPLE для моделирования на высокопроизводительных ЭВМ импульсной магнитоускоренной плазмы. // Математическое моделирование 24, с. 55–87, 2012.
9. Boldarev A., Gasilov V., Olkhovskaya O., Dyachenko S., Bagdasarov G., Boldyrev S., Gasilova I., Dorofeeva E. Object-oriented code MARPLE3D: simulations of radiative hydrodynamic/MHD effects at high-performance computer systems. In 6th European Congress on Computational Methods in Applied Sciences and Engineering, ECCOMAS 2012, Vienna, Austria, 10–14 September 2012, 2012.
10. Гасилов В.А., Болдарев А.С., Ольховская О.Г., Бойков Д.С., Шарова Ю.С., Савенко Н.О., Котельников А.М. MARPLE: программное обеспечение для мультифизического моделирования в механике сплошных сред // Вычислительные методы и программирование. 2023. 24, No 4. 316–338. https://doi.org/10.26089/NumMet.v24r423
11. Karypis, G. Karypis Lab https://github.com/KarypisLab
12. CMake. http://www.cmake.org/
13. Salome Platform. The open-source platform for numerical simulation. http://www.salome-platform.org/
14. ParaView. Open-source, multi-platform data analysis and visualization application. https://www.paraview.org/
15. AztecOO, Trilinos. https://trilinos.github.io/aztecoo.html
16. Heroux, M. A. AztecOO User Guide. Sandia National Laboratories, 2007. https://trilinos.github.io/pdfs/AztecOOUserGuide.pdf
17. JSON for Modern C++. https://json.nlohmann.me/
18. Гасилова И.В. Моделирование диссипативных процессов в пористых средах с газогидратными отложениями: дис. – Ин-т прикладной математики им. М.В. Келдыша РАН, 2016.