2025, **26** (2), 192–207. doi 10.26089/NumMet.v26r214

doi 10.26089/NumMet.v26r214 УДК 004.272

Разработка иерархической модели суперкомпьютерных систем в проекте Algo500: компонента CompZoo

А. С. Антонов
Московский государственный университет имени М. В. Ломоносова,
Научно-исследовательский вычислительный центр, Москва, Российская Федерация
ORCID: 0000-0003-2820-7196, e-mail: asa@parallel.ru

А. А. Щербаков
Московский государственный университет имени М. В. Ломоносова,
Научно-исследовательский вычислительный центр, Москва, Российская Федерация
ORCID: 0009-0009-0798-7075, e-mail: s02230566@gse.cs.msu.ru

Аннотация: Проект Algo500 представляет собой цифровую платформу для совместного анализа алгоритмов и архитектур суперкомпьютеров. В статье подробно рассматривается одна из компонент платформы — CompZoo, предназначенная для хранения детализированных описаний архитектур суперкомпьютерных систем. Приведено формальное описание используемой модели данных, основанной на представлении суперкомпьютера в виде иерархии подсистем. Рассматривается процесс разработки компоненты, включающий ключевые этапы реализации: проектирование структуры данных, создание визуальных представлений в виде графа и таблицы, а также реализацию пользовательского интерфейса с поддержкой фильтрации, сортировки и сохранения пользовательских настроек. В завершение обсуждаются текущие ограничения модели, рассматриваются возможные способы ее расширения и формулируются направления дальнейшего развития.

Ключевые слова: Algo500, CompZoo, суперкомпьютер, алгоритм, архитектура суперкомпьютеров, рейтинг, иерархическая модель, проектирование компонентов, реализация, визуализация данных.

Благодарности: Исследование выполнено в рамках государственного задания МГУ имени М. В. Ломоносова.

Для цитирования: Антонов А.С., Щербаков А.А. Разработка иерархической модели суперкомпьютерных систем в проекте Algo500: компонента CompZoo // Вычислительные методы и программирование. 2025. **26**, № 2. 192–207. doi 10.26089/NumMet.v26r214.

Development of a Hierarchical Model of Supercomputer Systems in the Algo500 Project: CompZoo component

Alexander S. Antonov
Lomonosov Moscow State University, Research Computing Center, Moscow, Russia
ORCID: 0000-0003-2820-7196, e-mail: asa@parallel.ru

Anton A. Shcherbakov
Lomonosov Moscow State University, Research Computing Center, Moscow, Russia
ORCID: 0009-0009-0798-7075, e-mail: s02230566@gse.cs.msu.ru

Abstract: The Algo500 project is a digital platform for the joint analysis of algorithms and supercomputer architectures. The article discusses in detail one of the components of the platform,

2025, **26** (2), 192–207. doi 10.26089/NumMet.v26r214

CompZoo, designed to store detailed descriptions of supercomputer system architectures. A formal description of the underlying data model is provided, based on representing a supercomputer as a hierarchy of subsystems. The article also describes the process of component development, including key implementation stages: data structure design, generation of visual representations in the form of graphs and tables, and the implementation of a user interface supporting filtering, sorting, and saving user settings. Finally, current limitations of the model are discussed, possible directions for its extension are considered, and further development plans are outlined.

Keywords: Algo500, CompZoo, supercomputer, algorithm, supercomputer architecture, rating, hierarchical model, component design, implementation, data visualization.

Acknowledgements: The research was carried out within the framework of the state assignment of Lomonosov Moscow State University.

For citation: A. S. Antonov, A. A. Shcherbakov, "Development of a Hierarchical Model of Supercomputer Systems in the Algo500 Project: CompZoo component," Numerical Methods and Programming. **26** (2), 192–207 (2025). doi 10.26089/NumMet.v26r214.

1. Введение. Алгоритмы лежат в основе программирования и вычислительной математики, определяя эффективность решения практических задач. Для их выполнения необходимы компьютеры, и особого внимания заслуживают суперкомпьютеры, которые способны решать сложные научные и инженерные задачи. Изучение их архитектуры помогает лучше понимать принципы работы этих систем и эффективно реализовывать алгоритмы для их использования.

Понимание этого привело к созданию проекта масштабируемой цифровой платформы Algo500, который направлен на решение задачи совместного анализа свойств алгоритмов и особенностей архитектур суперкомпьютеров [1]. Проект Algo500 призван совместно решать множество задач, таких как:

1) объединять данные о любых алгоритмах и архитектурах компьютеров,
2) позволять проводить анализ свойств любого алгоритма применительно к любой архитектуре суперкомпьютера на основании общего для всех алгоритмов подхода,
3) предоставлять возможность пользователям коллективно дополнять и уточнять базу данных алгоритмов и их реализаций,
4) вносить динамические характеристики выполнения реализаций алгоритмов на различных вычислительных системах,
5) генерировать по запросу различные списки, ранжированные по параметрам, задаваемым в платформе Algo500.

Структура платформы Algo500, основанная на подсистемах хранения данных, на сегодняшний день включает четыре взаимосвязанные компоненты: AlgoWiki, CompZoo, PerfData и RatingLists [1].

Данная работа посвящена разработке компоненты CompZoo, которая будет подробнее рассмотрена ниже. Прежде чем перейти к ее подробному описанию, необходимо дать общее представление о структуре проекта Algo500, охарактеризовав все его основные компоненты. Эти компоненты неоднократно описывались в различных научных статьях [1–6]:

1. AlgoWiki — наиболее проработанная компонента платформы Algo500. Это открытая энциклопедия свойств алгоритмов, основанная на движке MediaWiki, представленная в виде веб-сайта [7–9]. AlgoWiki предназначена для создания, хранения, публикации описаний алгоритмов, их свойств и реализаций для различных классов архитектур суперкомпьютеров [3, 4]. Важным является способ классификации различных алгоритмов, описания которых содержатся в AlgoWiki. Для этого используется схема "Задачи–Методы–Алгоритмы–Реализации", которая позволяет выстроить иерархию описания алгоритмов [1, 5, 6, 10]. Так, начиная с общего, абстрактного уровня описания задачи, осуществляется переход уже к конкретным методам и алгоритмам для ее решения, а затем и к конкретным реализациям этих алгоритмов.
2. CompZoo — компонента, предназначенная для хранения структурированных описаний архитектур суперкомпьютеров. Важной характеристикой этой компоненты является то, что она содержит более подробное описание архитектур суперкомпьютеров, чем в известных суперкомпьютерных рейтингах,

2025, **26** (2), 192–207. doi 10.26089/NumMet.v26r214

таких как Top500 или Graph500 [11, 12]. Это позволяет проводить более полный анализ параллельных свойств реализаций алгоритмов при их запуске, в частности, на подмножестве вычислительных узлов, входящих в состав суперкомпьютера [13].

3. Компонента PerfData — репозиторий данных, в котором наряду с программной реализацией алгоритмов и конфигурационными файлами для конкретной вычислительной системы содержатся сведения о выбранной для запуска программы совокупности вычислительных узлов суперкомпьютера, данные о входных аргументах, параметрах и результатах прогона программ [6].
4. Компонента RatingLists предназначена для построения рейтинговых списков, она дает пользователям возможность конструировать запросы к базам данных и представляет в браузере результаты этих запросов. Суть компоненты заключается именно в визуализации рейтингов — представлении их, в частности, в табличном виде и на графике, а данные для составления рейтингов берутся из базы данных компоненты PerfData, а также из CompZoo [1].

Далее вместо устоявшегося термина “компонента” применительно к AlgoWiki, CompZoo, PerfData и RatingLists мы будем применять термин “подсистема” с целью исключить конфликт, вызванный использованием в статье выражения “диаграмма компонентов” языка UML.

Ниже на рис. 1 представлена схема структуры проекта Algo500. Важно отметить, что подсистемы CompZoo и AlgoWiki в проекте являются самодостаточными, т.е. они не зависят от других подсистем. В то же время подсистема PerfData обращается к моделям данных подсистем CompZoo и AlgoWiki, а подсистема RatingLists — к PerfData, а значит, и ко всем трем другим подсистемам проекта.

Данная работа посвящена разработке подсистемы CompZoo. CompZoo является очень важной подсистемой, именно она предоставляет данные об архитектурах суперкомпьютеров, используемые для комплексного совместного анализа пользователями как самих архитектур суперкомпьютеров, так и результатов запусков суперкомпьютерных экспериментов, хранимых в подсистеме PerfData, с помощью инструментов подсистемы RatingLists.

2. Обзор предметной области. Современные рейтинги суперкомпьютеров, в том числе международный Top500, публикуемый с 1993 г., играют важную роль в отслеживании глобальных тенденций в области высокопроизводительных вычислений. Они включают ключевые характеристики систем: позицию в рейтинге, количество вычислительных ядер, пиковую и максимальную производительность, тип межсоединений, программное обеспечение и энергопотребление. Однако структура этих описаний ориентирована прежде всего на сравнительную оценку систем по результатам конкретного теста, такого как HPL (High Performance Linpack), и не отражает глубокой детализации архитектурных особенностей.

Региональные инициативы, например рейтинг Top50, охватывающий Россию и ряд соседних стран, используют более структурированную модель представления, включая информацию о конфигурациях

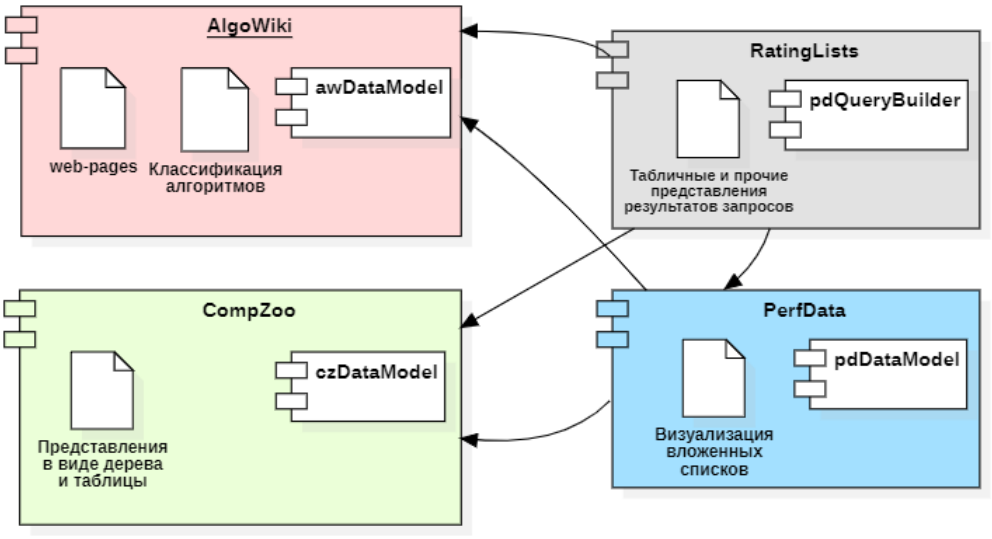


Рис. 1. Схема взаимодействия подсистем проекта Algo500

Fig. 1. Interaction diagram of the Algo500 project subsystems

2025, **26** (2), 192–207. doi 10.26089/NumMet.v26r214

узлов, типах процессоров, межсоединениях и их топологиях [14]. Тем не менее даже в этом случае описание ограничивается фиксированным набором атрибутов, без учета сложной иерархии памяти, внутриузловых связей и других значимых для анализа элементов архитектуры суперкомпьютеров.

Существуют и специализированные рейтинги, такие как IO500 (производительность ввода-вывода), Graph500 (работа с памятью и графовыми структурами), MLPerf (вычисления в области машинного обучения), которые оценивают отдельные аспекты вычислительных систем. Однако они не ставят целью формальное описание архитектуры суперкомпьютера в целом, а концентрируются на результатах выполнения узкоспециализированных тестов.

Несмотря на разнообразие рейтингов и систем учета, на текущий момент отсутствует единая общепринятая модель описания суперкомпьютерных архитектур, позволяющая:

- формализованно сравнивать архитектуры различных систем,
- отражать уникальные особенности конкретных реализаций,
- масштабировать описание по мере появления новых решений в архитектуре суперкомпьютерных систем.

Таким образом, несмотря на наличие обширных данных о конфигурации и производительности систем, их ограниченная структурная детализация затрудняет архитектурный анализ и сопоставление. Это подчеркивает необходимость создания формализованного и расширяемого подхода к описанию суперкомпьютеров, способного преодолеть ограничения существующих рейтингов. Одним из таких решений становится система CompZoo, разрабатываемая в рамках проекта Algo500, которая будет подробно рассмотрена в следующем разделе.

3. Модель данных подсистемы CompZoo.

3.1. Подход CompZoo к формализации архитектур. В ответ на выявленные ограничения существующих способов описания суперкомпьютеров в проекте Algo500 [15, 16] разрабатывается подсистема CompZoo, ориентированная на формальное и вариативное описание архитектур суперкомпьютеров. В основе CompZoo лежит иерархическая и расширяемая модель, которая позволяет описывать архитектуру на различных уровнях: от всей системы в целом до отдельных подсистем, таких как вычислительные узлы, процессоры или кэш-память. Для каждого уровня могут быть заданы как универсальные, так и специфические характеристики, определяющие свойства или состав элемента.

Такой подход обеспечивает:

1) гибкость в выборе набора характеристик и глубины описания,
2) поддержку расширяемости модели при изменении требований или появлении новых архитектурных решений,
3) возможность формального сопоставления систем на основании детального представления их конфигурации.

Таким образом, CompZoo предлагает независимую альтернативу существующим системам описания, акцентируя внимание на архитектурной детализации и обеспечивая основу для дальнейших формальных и визуальных методов анализа [17].

3.2. Формальное описание структуры подсистемы CompZoo. Суперкомпьютер — сложная техническая система, для упрощения описания которой исследователь вынужден перейти к модельным представлениям о суперкомпьютерах и сохранять в CompZoo редуцированные их описания в рамках некоторого набора характеристик выбранной модели.

Для CompZoo были рассмотрены многие модели, но в итоге была выбрана модель состава системы [1]. Данная модель представляет собой список подсистем, образованный при выделении в системе иерархии подсистем необходимой степени вложенности. Она представляет собой некоторое подмножество подсистем, представленных в спецификации суперкомпьютера. В CompZoo такое подмножество подсистем определяется целями исследования, причем эти подсистемы могут и не быть выделены явно в спецификации.

Формальное описание начинается с введения множества моделей суперкомпьютеров:

$$U_m = \{x : P(F(x))\}. \tag{1}$$

Здесь x — переменная, обозначающая конкретную модель суперкомпьютера; $F = \langle f_1, f_2, \ldots, f_N \rangle$ — кор-

2025, **26** (2), 192–207. doi 10.26089/NumMet.v26r214

теж предметных функторов, определенных на U_m, каждый из которых сопоставляет модели некоторое значение ее характеристики (например, пиковая производительность, год ввода в эксплуатацию, тип используемых процессоров и т.д.). Предикат P определяет условие истинности описания модели x путем сопоставления значений функторов заданным значениям характеристик:

$$P(x) = \bigwedge_{i=1}^{N} p_i(f_i(x), \beta_i), \quad \beta_i \in B_i. \tag{2}$$

Здесь p_i — бинарный предикат, например равенство или сравнение, который сравнивает значение функтора $f_i(x)$ с заданной константой β_i, B_i — множество допустимых значений i-й характеристики. Совокупность всех B_i образует множество B, охватывающее все множество значений, используемых в модели для описания характеристик суперкомпьютеров.

Таким образом, множество U_m содержит все возможные модели суперкомпьютеров, представимые в системе CompZoo в терминах предметных функторов и логических условий.

Каждому суперкомпьютеру x_k, где $k \in [1, |U_m|]$, сопоставляется подграф G^{x_k}, входящий в состав общего ориентированного ациклического графа $G^s = (V^s, E^s)$. Граф G^s представляет совокупность всех систем и подсистем, представленных в CompZoo, а также отношения иерархического включения между ними. Множество вершин подграфа G^{x_k} соответствует подсистемам данной модели:

$$V^{x_k} = \{v_i^{x_k} : P_i^{x_k}(F_i^{x_k}(v_i^{x_k}))\}, \quad i = 1, \ldots, |V^{x_k}|. \tag{3}$$

Здесь каждая вершина $v_i^{x_k}$ представляет собой конкретную подсистему суперкомпьютера x_k, описанную собственным набором функторов $F_i^{x_k}$ и предикатом $P_i^{x_k}$. Такая декомпозиция позволяет перейти от описания всей системы к формальному описанию ее составных частей с возможностью задания контекста и уровня детализации для каждой подсистемы.

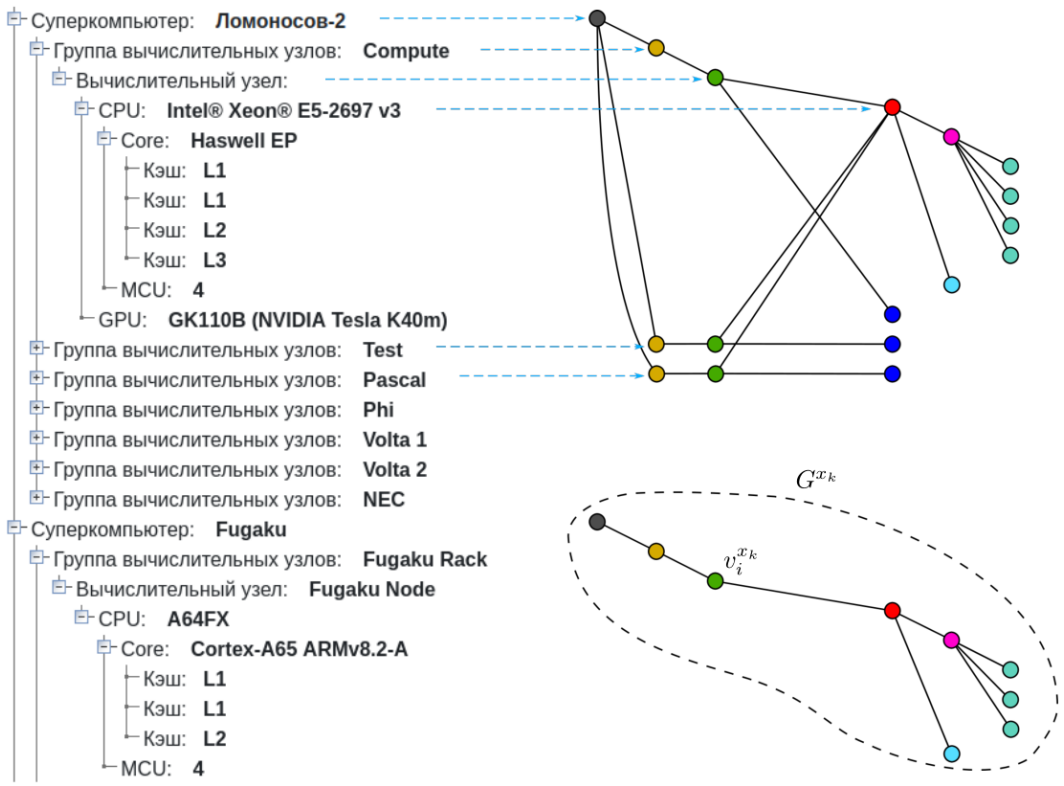


Рис. 2. Представление фрагментов моделей состава систем для суперкомпьютеров Ломоносов–2 и Fugaku: a) в виде списка подсистем в CompZoo; b) в виде графа

Fig. 2. Representation of fragments of system composition models for Lomonosov–2 and Fugaku supercomputers: a) as a list of subsystems in CompZoo; b) in the form of a graph

2025, **26** (2), 192–207. doi 10.26089/NumMet.v26r214

На рис. 2 приведены примеры описания суперкомпьютеров Ломоносов–2 и Fugaku в подсистеме CompZoo [18–20]. Рис. 2 a отражает модель состава системы, реализованную в виде иерархии подсистем, используемой в CompZoo. Рис. 2 b показывает соответствующий подграф G^{x_k}, включающий связи между подсистемами. Пунктирные стрелки показывают соответствие между представлениями.

Чтобы сохранить семантику вложенности и принадлежности, каждой подсистеме сопоставляется тег, определяющий ее роль в архитектуре (например, “суперкомпьютер”, “группа узлов”, “CPU” и др.). Тег формально задается функцией $T^{x_k}(v_i^{x_k})$, возвращающей строковое обозначение типа подсистемы. На рис. 2 a первыми отображаются теги подсистем, за которыми следует наименование соответствующей подсистемы, выделенное полужирным шрифтом. Название подсистемы формируется с использованием одноместного функтора f_1. Далее можно определить множество характеристик, объединенных по тегу tag, следующим образом:

$$C_{\text{tag}} = \bigcup_{\substack{x_k \in U_m \\ 1 \leqslant i \leqslant |V^{x_k}|}} F_i^{x_k} \quad \text{таких, что} \quad T^{x_k}(v_i^{x_k}) = \text{tag}. \tag{4}$$

Множество C_{tag} в модели CompZoo именуется категорией и представляет собой объединенный набор всех предметных функторов, используемых для описания подсистем с тегом tag во всех моделях суперкомпьютеров. Такая категория формирует единое пространство признаков, на основании которого возможно построение согласованного представления характеристик всех подсистем данного типа — в том числе в табличной форме. Это обеспечивает структурную совместимость данных, несмотря на вариативность и неполноту отдельных описаний.

Следует учитывать, что некоторые функторы могут быть заданы не для всех подсистем с данным тегом. В таких случаях значения соответствующих предикатов считаются неопределенными, что важно при формировании обобщенного набора признаков и последующей обработке.

Таким образом, категория C_{tag} представляет собой универсальную структурную единицу модели, обеспечивающую масштабируемость и сопоставимость описаний подсистем суперкомпьютеров. Это позволяет CompZoo адаптироваться к росту разнообразия архитектур и поддерживать обобщенную и формализованную основу для последующего анализа, визуализации и сравнения конфигураций.

4. Архитектурные особенности реализации Algo500. В данном разделе рассматриваются архитектурные и программные особенности реализации проекта Algo500, в частности механизмы клиент-серверного взаимодействия, организация классов, а также подходы к визуализации архитектурных данных в подсистеме CompZoo. Целью визуализации является представление архитектурных характеристик суперкомпьютеров в удобной форме для анализа и сравнения.

4.1. Изолированное представление. Реализация визуализации основана на выполнении вычислительных операций на стороне клиента. Такой подход выбран в связи с необходимостью поддержки механизма изолированного представления, применяемого в рамках проекта Algo500 для подсистем AlgoWiki, CompZoo и PerfData [1]. Согласно данной концепции, каждый пользователь должен иметь возможность видеть результат своих изменений, даже если они еще не были подтверждены модератором.

В подсистеме CompZoo используется СУБД PostgreSQL для хранения архитектур суперкомпьютеров, представимых в виде графов, хранящихся в формате JSON. Все действия пользователей, направленные на изменение структуры графа (добавление, удаление или модификация вершин и ребер), сохраняются в отдельном журнале изменений с привязкой к пользователю и статусом операции.

При входе пользователя в систему сервер считывает из базы данных:

- актуальный общий граф (глобальное представление без учета пользовательских изменений),
- все записи журнала изменений, относящиеся к данному пользователю и ожидающие подтверждения модератором.

При этом никаких модификаций графа на стороне сервера не производится. Вместо этого оба объекта (исходный граф и персональный журнал) передаются на клиентскую сторону, где осуществляется логическая обработка: данные из журнала применяются к структуре графа в оперативной памяти браузера, формируя тем самым изолированное представление, доступное только пользователю.

Обработка данных на стороне клиента обусловлена соображениями масштабируемости и производительности:

- пользователь получает возможность немедленно увидеть результат своих действий, без задержек, вызванных серверной обработкой,

2025, **26** (2), 192–207. doi 10.26089/NumMet.v26r214

- сервер разгружается от необходимости пересчета индивидуального представления для каждого пользователя, что критически важно при большом числе одновременно активных сессий,
- модель остается устойчивой к росту объемов данных и количеству пользователей, поскольку основная вычислительная нагрузка перемещается на клиента.

В отличие от традиционных механизмов транзакционной изоляции в СУБД, изолированное представление, реализованное в подсистемах Algo500, включает в себя не только логические изменения, но и модификацию визуальной структуры графа, что требует специализированной логики. Расчет этой логики на стороне клиента позволяет выполнять ее в интерактивном режиме с минимальной задержкой и полной независимостью между сессиями разных пользователей.

4.2. Клиент-серверная архитектура проекта Algo500. Важным программным компонентом в Algo500 является Message Queue Server (MQSRV), написанный на JavaScript для среды исполнения Node.js, в состав которого входят серверы, реализующие обмен данными с клиентами по протоколам HTTP и WebSocket. MQSRV в свою очередь взаимодействует с PostgreSQL-сервером, в котором находится БД проекта Algo500.

Клиент посредством JavaScript-программы из браузера взаимодействует с Message Queue Server. Также клиентская JavaScript-программа взаимодействует через Apache Web Server с сервером MediaWiki. В рамках данной работы важно именно взаимодействие с MQ-сервером для получения данных из CompZoo, хранящейся в БД Algo500.

Соответствующая диаграмма компонентов показана на рис. 3.

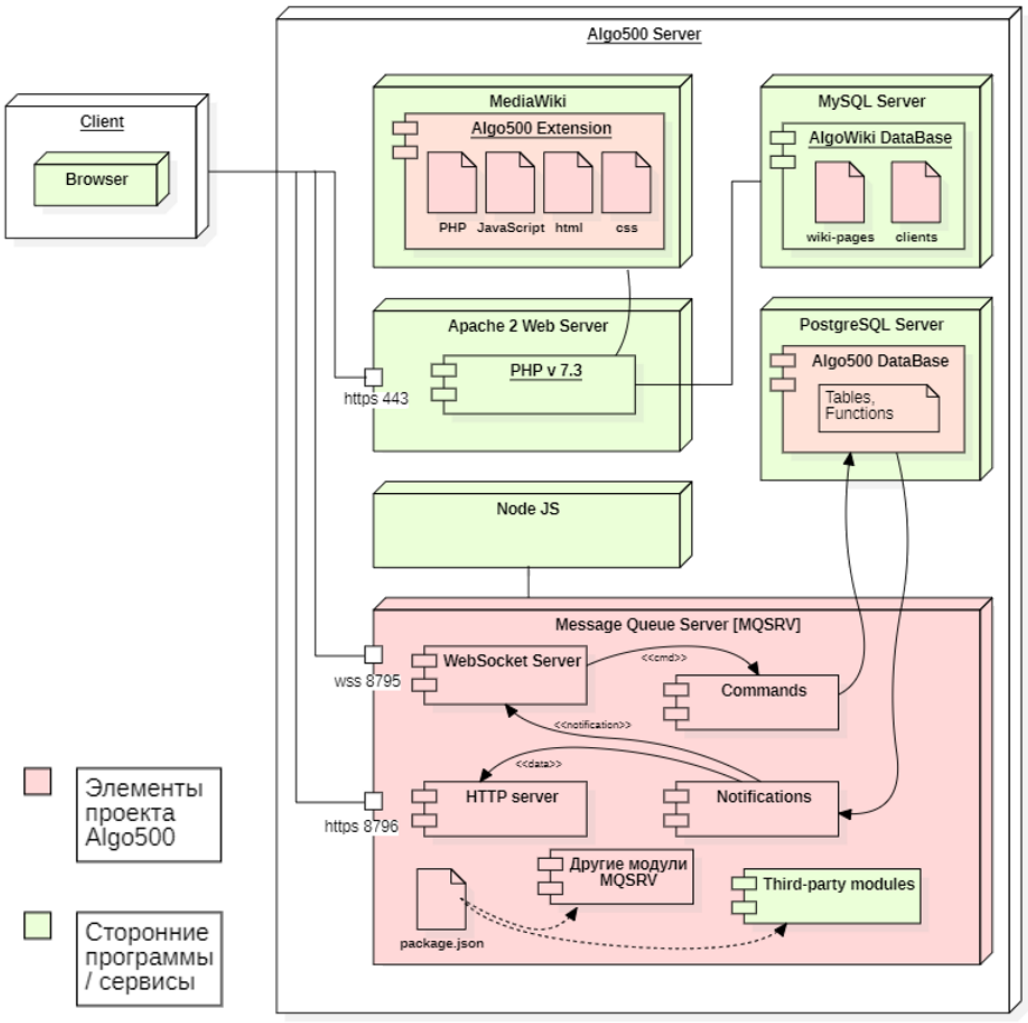


Рис. 3. Диаграмма компонентов Algo500

Fig. 3. Algo500 Components Diagram

2025, **26** (2), 192–207. doi 10.26089/NumMet.v26r214

4.3. Иерархия классов Algo500. Для разработки проекта Algo500, который представляет собой масштабную систему, было принято решение ограничить использование языка программирования JavaScript объектно-ориентированным подходом (ООП), учитывая его способность эффективно организовывать сложные структуры и обеспечивать гибкость в развитии системы. В рамках этого подхода в проекте была сформирована диаграмма классов, что является традиционной и хорошо зарекомендовавшей себя практикой в области программной инженерии.

Рассмотрим общую идею устройства этой диаграммы.

- Классы, представляющие подсистемы AlgoWiki, CompZoo, PerfData и RatingLists, наследуют от класса `Component` возможности по интеграции уникальных для каждой из подсистем классов: 1) класса `Menu`, содержащего интерфейс к функциям пользователя; 2) класса `AnalyticToolSet`, представляющего коллекцию инструментов для анализа данных.
- Набор инструментов `AnalyticToolSet` агрегирует классы `AnalyticTool`, представляющие конкретные инструменты для анализа данных, и дает пользователю возможность размещать на экране окна различных классов-наследников `AnalyticTool`.
- Классы, представляющие инструменты для анализа (наследники от `AnalyticTool`), адаптируются под задачи каждого компонента. Например, для CompZoo предусмотрены табличное и древовидные представления данных.

Таким образом, образуется механизм расширения системы как статически — через написание элементов программы, так и динамически — в процессе работы организовать добавление существующих `AnalyticTool` в выбранный `AnalyticToolSet`.

4.4. Исследование возможности применения библиотек JavaScript для визуализации. Для подсистемы CompZoo наиболее релевантными форматами визуализации архитектур суперкомпьютеров были признаны таблица и дерево — они соответствуют логике хранения данных и позволяют отразить как иерархию подсистем, так и внутреннюю их структуру и атрибуты отдельных элементов. Эти представления формируются на стороне клиента посредством использования связки JavaScript, CSS и HTML. Для их вывода на странице создаются наборы инструментов `AnalyticToolSet` и сами инструменты `AnalyticTool`.

При выборе подходов к представлению и выводу данных рассматривалась возможность использования сторонних JavaScript-библиотек, способных обеспечить интерактивность, широкий набор виджетов для вывода данных, включая готовые компоненты для табличных и иерархических представлений, а также средств для формирования отзывчивого интерфейса и эффективного решения базовых задач визуализации.

Были рассмотрены такие библиотеки визуализации, как Webix, EasyUI, w2ui и др. В качестве основы для тестового решения и оценки функциональных возможностей была выбрана библиотека Webix [21], поскольку она предоставляет широкий набор инструментов и шаблонов для визуализации, включая такие ключевые виджеты, как DataTable [22] и Tree [23], которые необходимы для реализации проекта Algo500.

Дополнительным фактором в пользу выбора библиотеки Webix стало то, что она не требует использования JavaScript-фреймворков и функционирует на чистом JavaScript. В проекте Algo500 отказ от фреймворков обусловлен тем, что они, как правило, ориентированы на решение типовых задач с реляционным представлением данных. В условиях же проекта, предполагающего работу с нетипичными структурами, применение таких технологий может привести лишь к усложнению разработки.

С помощью библиотеки Webix была сконфигурирована структура графа и таблицы и произведена их визуализация.

Однако построенное решение позволило наглядно выявить ряд существенных ограничений:

1. Ограниченная настройка. Кастомизация элементов в библиотеках типа Webix часто трудоемка и требует тщательного изучения документации, а некоторые элементы вовсе могут не поддаться корректировке.
2. Сложность освоения. Для работы с Webix необходимо изучить множество аспектов, что требует времени, особенно если проект коллективный.
3. Ограниченный функционал без лицензии. Бесплатная версия Webix имеет ограничения, что может препятствовать реализации нужных функций, например объединению ячеек таблицы.

Вспомогательные JavaScript-библиотеки могут быть полезны для множества проектов, однако для специфических задач, как в проекте Algo500, их использование имеет существенные ограничения. Несмот-

2025, **26** (2), 192–207. doi 10.26089/NumMet.v26r214

ря на определенную возможность кастомизации элементов, достижение нужного внешнего вида, в частности для таблицы оказалось невозможным.

В итоге, несмотря на привлекательность ряда готовых решений, было принято решение реализовывать визуализацию исключительно на чистом JavaScript. Такой подход обеспечивает максимальную гибкость, совместимость с архитектурой проекта и возможность учесть специфические требования к отображению и обработке архитектурных описаний.

5. Итоговое решение для отображения данных подсистемы CompZoo.

5.1. Структура формируемого решения. В предыдущем разделе был рассмотрен процесс формирования решения, который в некотором роде имел характер “снизу-вверх”. На этом этапе проводились исследовательские работы по выбору и апробации алгоритмов и библиотек. В частности, весь код, связанный с формированием таблицы CompZoo, построением дополнительных представлений для вывода, считыванием данных из БД, был реализован в нескольких функциях одного класса.

После принятия решения об используемом подходе была спроектирована структура программы с использованием метода “сверху-вниз”. Для этого было произведено разбиение кода на несколько логических частей, каждая из которых решает конкретную задачу. Такой подход обеспечил следующие преимущества: повышение читаемости и прозрачности программы; упрощение тестирования на уровне отдельных классов; легкость в модификации программы.

Для разделения функций была сформирована диаграмма классов, где были выделены 4 региона:

1. Классы, отвечающие за предварительные действия по получению данных из БД и их обработке.
2. Классы, конечной задачей которых является построение html-представления в виде дерева.
3. Классы, конечной задачей которых является построение html-представления в виде таблицы.
4. Регион, объединяющий обозначенные представления в инструменты анализа AnalyticTool, формирующих AnalyticToolSet.

В каждом регионе имеется множество классов, разбивающих логику построения, например html-дерева на более мелкие части. Так, например, есть классы, устанавливающие обработчики событий на элементы html-дерева, классы, отвечающие за вывод данных в нужном формате в ячейках графа, за разделение экранной области между окнами и др.

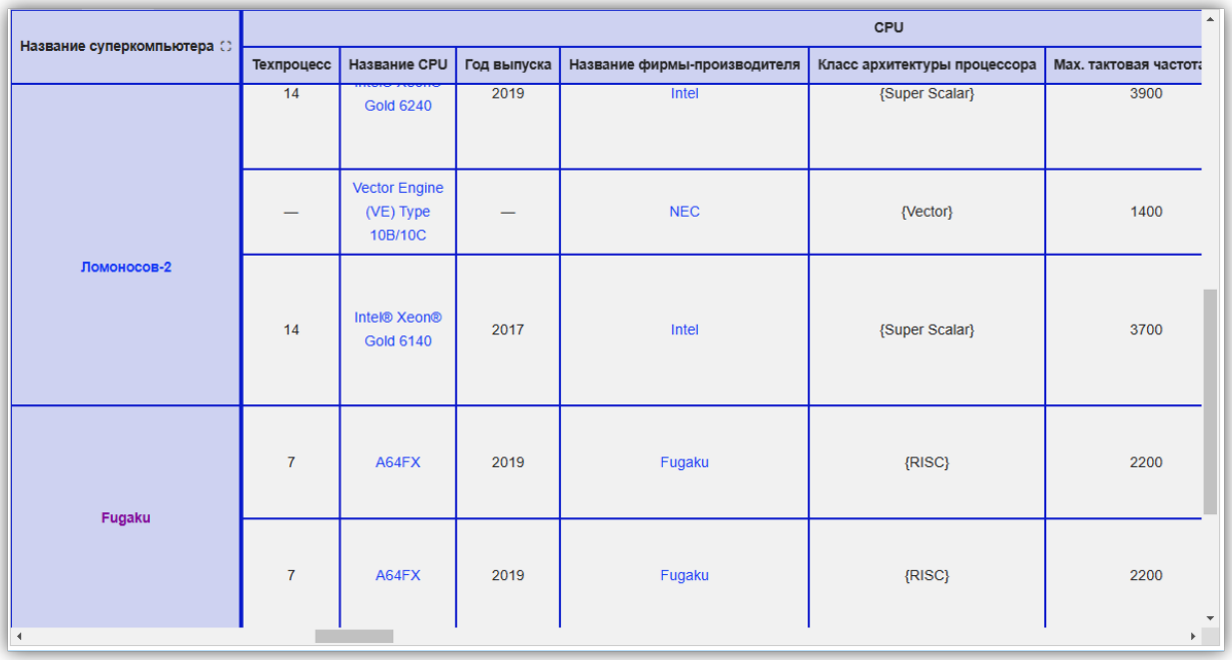
В разделе 4 была кратко описана ООП-структура Algo500. И таким образом, формируемые в решении инструменты визуализации в табличном и графовом видах в AnalyticTool успешно встраиваются в проект Algo500.

5.2. Получение данных из БД. Получение данных из БД CompZoo клиентов ведется посредством взаимодействия с компонентом Message Queue Server проекта Algo500. Для этого в серверной части реализована поддержка протоколов HTTP и WebSocket. Также написан WebSocket-клиент, который устанавливает соединение с сервером обмена сообщениями. Соответственно, сервер считывает нужные таблицы БД CompZoo и присылает их клиенту. Далее на клиентской стороне идет процесс построения изолированного представления. Для этого посредством считанной таблицы `journal` из БД идет редактирование основного графа, содержащего все суперкомпьютерные системы, доступные в БД CompZoo. В результате клиентская программа JavaScript располагает всеми необходимыми данными для дальнейшей работы.

Далее с использованием считанных данных формируются дополнительные представления для работы следующих в иерархии классов компонентов. Так, формируются Map-структуры с идентификаторами узлов подсистем и самими подсистемами, Map-структуры, формирующие категорию подсистем по ее идентификатору, также Map-структуры для хранения дочерних элементов и тегов текущих элементов и некоторые другие. Эти структуры передаются далее по веткам иерархии в вышележащие классы, посвященные построению графа и таблицы.

5.3. Отображение данных о суперкомпьютерах в табличном виде. Для формирования таблицы вначале строится промежуточное представление в виде двумерного массива. Строками массива являются будущие строки таблицы. И каждая строка массива, в свою очередь, тоже представляет собой массив — массив объектов. Каждый объект в этом массиве представляет собой подсистему с атрибутами (и тегом), которые потом будут надлежащим образом считаны в расположенном выше классе, и в конечном счете будет построена html-таблица. Не будем подробно описывать процесс создания промежуточного представления таблицы из графа, рассмотрим лишь общую идею: она заключается в проходе “в ширину” (алгоритм BFS) по полученному графу с созданием необходимых rowspan-атрибутов, возмож-

2025, **26** (2), 192–207. doi 10.26089/NumMet.v26r214



Название суперкомпьютера	CPU					
	Техпроцесс	Название CPU	Год выпуска	Название фирмы-производителя	Класс архитектуры процессора	Max. тактовая частота
Ломоносов-2	14	Gold 6240	2019	Intel	{Super Scalar}	3900
	—	Vector Engine (VE) Type 10B/10C	—	NEC	{Vector}	1400
	14	Intel® Xeon® Gold 6140	2017	Intel	{Super Scalar}	3700
Fugaku	7	A64FX	2019	Fugaku	{RISC}	2200
	7	A64FX	2019	Fugaku	{RISC}	2200

Рис. 4. Итоговый вид таблицы (показана частично)

Fig. 4. The final view of the table (partially shown)

ным добавлением фиктивных подсистем для сохранения структуры таблицы, заполнением значениями полей.

Пример итогового отображения части таблицы приведен на рис. 4. Прочерки в некоторых ячейках таблицы соответствуют отсутствующим атрибутам или целым подсистемам на соответствующем уровне вложенности и являются результатом выравнивания структуры графа при преобразовании в таблицу с фиксированным набором атрибутов.

5.4. Отображение данных о суперкомпьютерах в графовом виде. Для формирования решения в виде графа вначале формируется его “сырая” структура. Идея этой структуры была взята с веб-сайта [24]. Каждый узел `Node` в дереве состоит из иконки `Expand`, заголовка `Content` и контейнера для дочерних элементов `Container`. Посредством прохождения по полученному графу из БД и заполнения нужными значениями данных полей и формируется базовая структура дерева.

Далее над этой структурой посредством работы других классов производятся дополнительные надстройки: добавляются обработчики выбора узла дерева с выводом атрибутов в отдельной таблице, строится полная таблица, начиная от выделенного узла в дереве (по аналогии с таблицей, описываемой выше). Кроме того, добавляются функции сортировки и фильтрации элементов дерева.

Для повышения удобства и персонализации работы пользователя реализован механизм сохранения состояния пользовательского интерфейса с использованием объекта `LocalStorage` браузера. В частности, сохраняются: состояние раскрытых узлов дерева, параметры сортировки и фильтрации, а также настройки компоновки элементов интерфейса. При повторном открытии страницы эти параметры автоматически восстанавливаются, что позволяет пользователю продолжать работу с системой в привычной конфигурации без необходимости повторной настройки интерфейса. Такая реализация существенно повышает интерактивность решения и улучшает пользовательский опыт.

Итоговый вид отображения дерева показан на рис. 5.

6. Ограничения CompZoo и направления развития.

6.1. Проблемы и ограничения CompZoo. Несмотря на обозначенные в разделе 3 преимущества модели CompZoo, она также имеет ряд проблем и ограничений [1]:

1. Проблема описания иерархии подчиненности между подсистемами. Одной из ключевых трудностей при описании суперкомпьютеров в принятой модели CompZoo является корректное отражение количества однотипных подсистем. При описании суперкомпьютерной системы часто необходимо указать

2025, **26** (2), 192–207. doi 10.26089/NumMet.v26r214

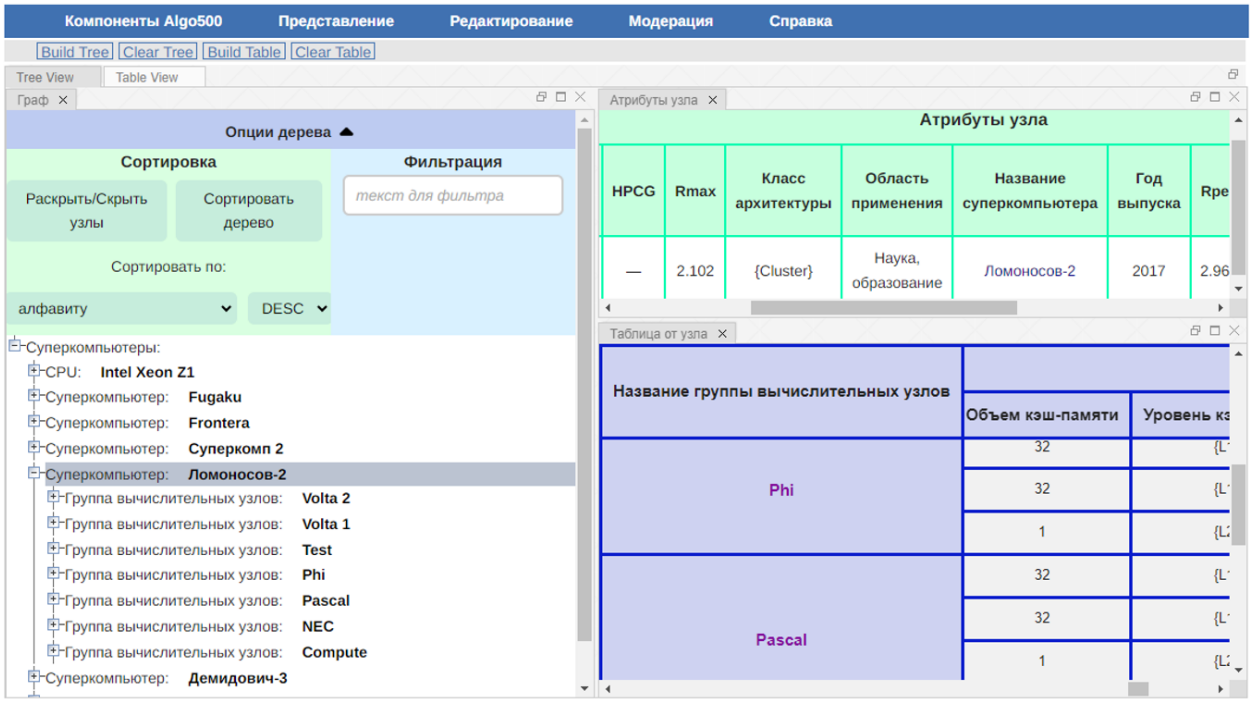


Рис. 5. Интерфейс графового представления архитектур

Fig. 5. Graph-based interface for supercomputer architectures

количество подсистем, принадлежащих ей. Это может быть, в частности, 160 вычислительных узлов в составе группы узлов, или 8 CPU и 8 GPU в составе узла. При структурированном представлении данных возникает вопрос: указывать количество подсистем в характеристиках более крупной группы или непосредственно в описании самой подсистемы.

Рассмотрим первый вариант — фиксировать количество компонентов на уровне вышестоящей подсистемы. Такой подход схож с принципами конструкторской документации, где спецификации верхнего уровня содержат информацию о количестве вложенных элементов. Однако в некоторых описаниях суперкомпьютеров с кластерной архитектурой выделенной сущности “группа вычислительных узлов” может не существовать (например, система Frontier [25]). В таких случаях исследователь оказывается перед выбором: либо вводить эту подсистему искусственно, либо работать с неявными ограничениями, что усложняет процесс сравнения различных систем.

Второй вариант — фиксировать количественные характеристики непосредственно в описании отдельных узлов. Однако этот подход плохо сочетается с графовой моделью, так как затрудняет многократное использование типовых подсистем с варьирующимся количеством экземпляров для разных суперкомпьютеров. В результате описание теряет гибкость и возможность репликации однотипных элементов оказывается ограниченной.

2. Проблема точности и согласованности описаний. Разработчики могут выделить различающиеся наименования подсистем, функционально очень схожих между собой. Так, “разделы” суперкомпьютера Ломоносов–2 [20, 26] и “primary compute system” в Frontera [27] логично объединить в подсистему с одним тегом, однако это может вызвать несогласие у части исследователей.

   Другой проблемой являются несовпадающие единицы измерения величин, которые не могут быть легко приведены друг к другу. Например, в контексте задержек обращений к памяти одни авторы могут использовать такты, когда другие — наносекунды.
3. Проблема стандартизации наименований единиц измерения. Проблема возникает из-за несоответствий в обозначениях таких величин, как flops, teps и др. Эти единицы могут иметь различные представления в разных источниках. В последние годы, в соответствии с рекомендациями международных стандартов МЭК (ИЭК), в зарубежных публикациях все чаще используются приставки типа “киби-”, “меби-” и др., обозначающие кратность степени числа 2. Однако в России применение этих стандартов не является обязательным, что приводит к расхождениям между отечественными

2025, **26** (2), 192–207. doi 10.26089/NumMet.v26r214

и международными практиками [28]. В лучшем случае российские публикации могут использовать обозначения, соответствующие ГОСТ 8.417–2002 [29], что также создает дополнительные трудности при сравнении данных. Для эффективного функционирования системы CompZoo необходимо внедрение четких соглашений по значениям и наименованиям единиц измерения.

6.2. Альтернативные подходы и возможные расширения модели CompZoo. В статье [15] рассматривается проблема построения иерархической модели для описания архитектур суперкомпьютеров. Проанализированы существующие подходы, а также предложен авторский метод описания архитектуры суперкомпьютеров.

1. Использование взвешенных ребер в графовой модели.

   В рассматриваемой статье предложен вариант описания суперкомпьютерных систем с использованием взвешенных ребер в графовой модели. Такой подход изменяет способ хранения информации о количественных характеристиках подсистем и предполагает, что веса ребер хранятся отдельно, а не в вершинах графа.

   Применение взвешенных ребер требует изменения принципов хранения данных, поскольку для описания системы необходимо выделять дополнительные структуры в базе данных, отвечающие за связи между подсистемами. Также пересмотру подлежит сам механизм взаимодействия компонентов модели, так как отношения между подсистемами в таком случае фиксируются через параметры связей, а не через атрибуты вершин.

2. Добавление веток графа для интерконнектов.

   В статье также предложено расширить графовую модель за счет включения отдельных веток, описывающих интерконнекты между подсистемами. Такие ветки предназначены для хранения информации о параметрах соединений между вычислительными узлами и могут содержать данные о пропускной способности каналов связи, латентности, а также топологии соединений между компонентами суперкомпьютера.

   Добавление подобных элементов в модель изменяет структуру описания системы, поскольку появляется необходимость фиксировать характеристики сетевых соединений наряду с описанием подсистем. В результате такой подход предполагает введение дополнительных объектов в модель, что требует учета этих особенностей при обработке данных и визуализации структуры системы.

3. Проблема описания атрибутов.

   В статье отмечена сложность, связанная с динамическим расширением атрибутов подсистем. Указано, что рост числа атрибутов приводит к увеличению объемов данных, необходимых для описания системы, и может усложнить их обработку. В частности, модель, в которой атрибуты подсистем задаются динамически, требует механизмов управления их изменяемостью, а также методов структурированной организации данных.

   С точки зрения представления информации, динамическое расширение атрибутов позволяет учитывать широкий спектр характеристик суперкомпьютеров, но при этом требует разработки подходов к оптимальному хранению и обработке данных. Вопрос заключается в том, каким образом учитывать эту изменяемость в модели, чтобы сохранить структурированность данных и удобство их использования в анализе.

Рассмотренные подходы предлагают альтернативные способы представления архитектуры суперкомпьютеров. Эти решения представляют собой возможные направления развития моделей описания суперкомпьютеров, однако их применение требует учета специфики и конкретных требований. В рамках проекта CompZoo сохраняется ориентированность на простоту и гибкость модели, что позволяет эффективно решать поставленные задачи. Рассмотренные альтернативные решения могут представлять интерес в контексте дальнейших исследований и возможных направлений развития.

6.3. Планы дальнейшей разработки и развития модели CompZoo. На текущем этапе реализовано решение, обеспечивающее визуальное представление архитектуры суперкомпьютеров в графовом и табличном форматах. Планы дальнейшей работы охватывают как развитие клиентской функциональности, так и расширение возможностей самой модели описания архитектур CompZoo.

Выделим следующие возможные направления дальнейшего развития:

- Доработка визуального интерфейса: добавление новых функций для анализа таблиц и графа, включая группировку по тегам, расширенную фильтрацию и сравнение подсистем.

2025, **26** (2), 192–207. doi 10.26089/NumMet.v26r214

- Разработка инструментов ввода и редактирования данных через интерфейсы в виде форм с возможностью их встраивания в веб-интерфейс проекта Algo500.
- Внедрение механизмов сравнения подсистем на уровне выбранных характеристик с возможностью построения аналитических таблиц и отчетов.
- Пересмотр и возможная адаптация структуры БД CompZoo на базе рассмотренных в разделе 6.2 идей альтернативных подходов: внедрения поддержки взвешенных связей между подсистемами, введения описания интерконнектов как отдельных объектов графа.

7. Заключение. В статье проведен анализ актуальности разработки компоненты CompZoo, детально описывающей архитектуру суперкомпьютерных систем. Произведено ее сравнение с моделями, использующимися в суперкомпьютерных рейтингах, таких как Top500, Top50, Graph500 и др. Рассмотрена принятая модель для описания суперкомпьютеров в виде состава системы. Подробно рассмотрена архитектура проекта Algo500 и дано обоснование целесообразности реализации большей части логики по созданию представлений в виде графа и таблицы CompZoo на стороне клиента.

Приведен процесс непосредственной разработки компоненты CompZoo, где описаны основные проделанные шаги, а именно: представлена структура иерархии классов решения, встраиваемая в проект Algo500; описан процесс построения html-кода таблицы, включающий создание промежуточного представления в виде двумерного массива и его дальнейшей интерпретации; дано описание процесса построения html-кода дерева, где также помимо базовой структуры решения представлены дополнительные функциональные надстройки, включающие интерфейс взаимодействия с возможностью фильтрации, сортировки и выделения узлов с последующим отображением сопутствующих данных, а также механизм сохранения пользовательского состояния посредством LocalStorage.

В завершение были выделены ключевые ограничения текущей модели, рассмотрены альтернативные подходы в рамках статьи [15] и представлены возможные направления дальнейшего развития компоненты CompZoo, включающие как совершенствование визуального интерфейса, так и расширение модели описания архитектур суперкомпьютеров. Эти направления открывают перспективы для повышения точности представлений, улучшения сравнимости конфигураций и дальнейшего развития функциональности.

Список литературы

1. *Антонов А.С., Майер Р.В.* Онтологический анализ предметной области цифровой платформы Algo500 // Вычислительные методы и программирование. 2023. **24**, № 1. 89–114. doi 10.26089/NumMet.v24r107.
2. *Antonov A.S., Nikitenko D.A., Voevodin Vl.V.* Algo500 — a new approach to the joint analysis of algorithms and computers // Lobachevskii J. Math. 2020. **41**, N 8. 1435–1443. doi 10.1134/S1995080220080041.
3. *Antonov A., Frolov A., Konshin I., Voevodin Vl.* Hierarchical domain representation in the AlgoWiki encyclopedia: from problems to implementations // Communications in Computer and Information Science. Vol. 910. Cham: Springer, 2018. 3–15. doi 10.1007/978-3-319-99673-8_1.
4. *Воеводин Вл.В.* Открытая энциклопедия свойств алгоритмов AlgoWiki: от мобильных платформ до экзафлопсных суперкомпьютерных систем // Вычислительные методы и программирование. 2015. **16**, № 1. 99–111. doi 10.26089/NumMet.v16r111.
5. *Popov A., Nikitenko D., Antonov A., Voevodin Vl.* Formal model of problems, methods, algorithms and implementations in the advancing AlgoWiki open encyclopedia // CEUR Workshop Proc. 2018. **2281**, 1–11. https://ceur-ws.org/Vol-2281/paper-01.pdf. Cited May 12, 2025.
6. *Antonov A.S., Maier R.V.* Development and implementation of the Algo500 scalable digital platform architecture // Lobachevskii J. Math. 2022. **43**, N 4. 837–847. doi 10.1134/S1995080222070058.
7. *Antonov A.* Wiki representation and analysis of knowledge about algorithms // Lecture Notes in Computer Science. Vol. 13708. Cham: Springer, 2022. 604–616. doi 10.1007/978-3-031-22941-1_44.
8. MediaWiki. https://www.mediawiki.org/wiki/MediaWiki. Cited May 12, 2025.
9. Алговики: открытая энциклопедия свойств алгоритмов. https://algowiki-project.org/ru/. Cited May 12, 2025.
10. *Antonov A.S., Dongarra J., Voevodin Vl.* AlgoWiki project as an extension of the Top500 methodology // Supercomput. Front. Innov. 2018. **5**, No. 1. 4–10. doi 10.14529/jsfi180101.
11. Home - | TOP500. https://www.top500.org. Cited May 12, 2025.
12. Graph 500 | large-scale benchmarks. https://graph500.org. Cited May 12, 2025.

2025, **26** (2), 192–207. doi 10.26089/NumMet.v26r214

13. *Antonov A., Voevodin Vad., Voevodin Vl., Teplov A.* A study of the dynamic characteristics of software implementation as an essential part for a universal description of algorithm properties // Proc. 24th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing. Heraklion, Greece, February 17–19, 2016. New York: IEEE Press, 2016. 359–363. doi 10.1109/PDP.2016.24.
14. Top50 | Суперкомпьютеры. http://top50.supercomputers.ru/list. Cited May 12, 2025.
15. *Nikitenko D.A.* Hierarchical model of architecture of supercomputer systems for comparison and ranking // Вестник ЮУрГУ. Серия "Вычислительная математика и информатика". 2022. **11**, № 4. 5–18. doi 10.14529/cmse220401.
16. *Nikitenko D., Antonov A., Zheltkov A., Voevodin Vl.* Describing HPC system architecture for understanding its capabilities // Communications in Computer and Information Science. Vol. 1331. Cham: Springer, 2020. 425–435. doi 10.1007/978-3-030-64616-5_37.
17. *Желтков А.А.* Разработка методов построения рейтингов вычислительных систем, основанных на реализациях различных алгоритмов // Суперкомпьютерные дни в России: Труды международной конференции, 23–24 сентября 2019. М.: МАКС Пресс, 2019. 192–199.
18. *Антонов А.С., Афанасьев И.В., Воеводин Вл.В.* Высокопроизводительные вычислительные платформы: текущий статус и тенденции развития // Вычислительные методы и программирование. 2021. **22**, № 2. 138–181. doi 10.26089/NumMet.v22r210.
19. About Fugaku | RIKEN Center for Computational Science. https://www.r-ccs.riken.jp/en/fugaku/about/. Cited May 12, 2025.
20. PARALLEL.RU. Суперкомпьютерный комплекс МГУ, включая суперкомпьютер "ЛОМОНОСОВ-2"; | PARALLEL.RU — Информационно-аналитический центр по параллельным вычислениям. https://parallel.ru/cluster/lomonosov2.html. Cited May 12, 2025.
21. Webix JS UI Library & Framework - JavaScript UI Widgets for Fast Web App Development. https://webix.com/. Cited May 12, 2025.
22. DataTable - DataTable UI widget documentation: configuration, data export, etc. Webix Docs. https://docs.webix.com/datatable__index.html. Cited May 12, 2025.
23. Tree, UI Widgets Webix Docs. https://docs.webix.com/datatree__index.html. Cited May 12, 2025.
24. Формирование дерева в JavaScript. https://javascript.ru/ui/tree. Cited May 12, 2025.
25. Frontier User Guide - OLCF User Documentation. https://docs.olcf.ornl.gov/systems/frontier_user_guide.html. Cited May 12, 2025.
26. *Voevodin Vl.V., Antonov A.S., Nikitenko D.A., et al.* Supercomputer Lomonosov-2: large scale, deep monitoring and fine analytics for the user community // Supercomput. Front. Innov. 2019. **6**, N 2. 4–11. doi 10.14529/jsfi190201.
27. System Architecture - TACC Frontera User Guide. https://frontera-portal.tacc.utexas.edu/user-guide/system/. Cited May 12, 2025.
28. Системы управления терминологией, базами знаний и контентом. Концептуальные аспекты разработки и интернационализации систем классификации: ГОСТ Р ИСО 22274–2016. М.: Стандартинформ, 2017.
29. Государственная система обеспечения единства измерений. Единицы величин: ГОСТ 8.417–2002. М.: Стандартинформ, 2019.

Поступила в редакцию
28 февраля 2025 г.

Принята к публикации
10 мая 2025 г.

Информация об авторах

Александр Сергеевич Антонов — к.ф.-м.н., вед. науч. сотр.; Московский государственный университет имени М. В. Ломоносова, Научно-исследовательский вычислительный центр, Ленинские горы, 1, стр. 4, 119991, Москва, Российская Федерация.

Антон Алексеевич Щербаков — студент магистратуры, Московский государственный университет имени М. В. Ломоносова, Научно-исследовательский вычислительный центр, Ленинские горы, 1, стр. 4, 119991, Москва, Российская Федерация.

References

1. A. S. Antonov and R. V. Maier, "Ontological Analysis of the Subject Area of the Algo500 Digital Platform," Numerical Methods and Programming **24** (1), 89–114 (2023). doi 10.26089/NumMet.v24r107.
2. A. S. Antonov, D. A. Nikitenko, and Vl. V. Voevodin, "Algo500 — a New Approach to the Joint Analysis of Algorithms and Computers," Lobachevskii J. Math. **41** (8), 1435–1443 (2020). doi 10.1134/S1995080220080041.

2025, **26** (2), 192–207. doi 10.26089/NumMet.v26r214

3. A. Antonov, A. Frolov, I. Konshin, and Vl. Voevodin, "Hierarchical Domain Representation in the AlgoWiki Encyclopedia: from Problems to Implementations," in *Communications in Computer and Information Science* (Springer, Cham, 2018), Vol. 910, pp. 3–15. doi 10.1007/978-3-319-99673-8_1.
4. Vl. V. Voevodin, "An Open AlgoWiki Encyclopedia of Algorithmic Features: from Mobile to Extreme Scale," Numerical Methods and Programming **16** (1), 99–111 (2015). doi 10.26089/NumMet.v16r111.
5. A. Popov, D. Nikitenko, A. Antonov, and Vl. Voevodin, "Formal Model of Problems, Methods, Algorithms and Implementations in the Advancing AlgoWiki Open Encyclopedia," CEUR Workshop Proc. **2281**, 1–11 (2018). https://ceur-ws.org/Vol-2281/paper-01.pdf. Cited May 12, 2025.
6. A. S. Antonov and R. V. Maier, "Development and Implementation of the Algo500 Scalable Digital Platform Architecture," Lobachevskii J. Math. **43** (4), 837–847 (2022). doi 10.1134/S1995080222070058.
7. A. Antonov, "Wiki Representation and Analysis of Knowledge about Algorithms," in *Lecture Notes in Computer Science* (Springer, Cham, 2022), Vol. 13708, pp. 604–616. doi 10.1007/978-3-031-22941-1_44.
8. MediaWiki. https://www.mediawiki.org/wiki/MediaWiki. Cited May 12, 2025.
9. Open Encyclopedia of Parallel Algorithmic Features — Algowiki. https://algowiki-project.org/en/. Cited May 12, 2025.
10. A. S. Antonov, J. Dongarra, and Vl. Voevodin, "AlgoWiki Project as an Extension of the Top500 Methodology," Supercomput. Front. Innov. **5** (1), 4–10 (2018). doi 10.14529/jsfi180101.
11. Home - | TOP500. https://www.top500.org. Cited May 12, 2025.
12. Graph 500 | large-scale benchmarks. https://graph500.org. Cited May 15, 2025.
13. A. Antonov, Vad. Voevodin, Vl. Voevodin, and A. Teplov, "A Study of the Dynamic Characteristics of Software Implementation as an Essential Part for a Universal Description of Algorithm Properties," in *Proc. 24th Euromicro Int. Conf. on Parallel, Distributed, and Network-Based Processing. Heraklion, Greece, February 17–19, 2016* (IEEE Press, New York, 2016), pp. 359–363. doi 10.1109/PDP.2016.24.
14. Top50 | Supercomputers. http://top50.supercomputers.ru/list. Cited May 12, 2025.
15. D. A. Nikitenko, "Hierarchical Model of Architecture of Supercomputer Systems for Comparison and Ranking," Vestn. YuUrGU. Ser. Vych. Matem. Inform. **11** (4), 5–18 (2022). doi 10.14529/cmse220401.
16. D. Nikitenko, A. Antonov, A. Zheltkov, and Vl. Voevodin, "Describing HPC System Architecture for Understanding Its Capabilities," in *Communications in Computer and Information Science* (Springer, Cham, 2020), Vol. 1331, pp. 425–435. doi 10.1007/978-3-030-64616-5_37.
17. A. A. Zheltkov, "Development of Methods for Constructing Ratings of Computing Systems Based on Implementations of Various Algorithms," in *Proc. Int. Conf. on Russian Supercomputing Days, Moscow, Russia, September 23–24, 2019* (MAKS Press, Moscow, 2019), pp. 192–199.
18. A. S. Antonov, I. V. Afanasyev, and Vl. V. Voevodin, "High-Performance Computing Platforms: Current Status and Development Trends," Numerical Methods and Programming **22** (2), 138–181 (2021). doi 10.26089/NumMet.v22r210.
19. About Fugaku | RIKEN Center for Computational Science. https://www.r-ccs.riken.jp/en/fugaku/about/. Cited May 12, 2025.
20. PARALLEL.RU: Supercomputer Lomonosov-2. https://parallel.ru/cluster/lomonosov2.html. Cited May 12, 2025.
21. Webix JS UI Library & Framework - JavaScript UI Widgets for Fast Web App Development. https://webix.com/. Cited May 12, 2025.
22. DataTable - DataTable UI widget documentation: configuration, data export, etc. Webix Docs. https://docs.webix.com/datatable__index.html. Cited May 12, 2025.
23. Tree, UI Widgets Webix Docs. https://docs.webix.com/datatree__index.html. Cited May 12, 2025.
24. Building tree in JavaScript. https://javascript.ru/ui/tree. Cited May 12, 2025.
25. Frontier User Guide - OLCF User Documentation. https://docs.olcf.ornl.gov/systems/frontier_user_guide.html. Cited May 12, 2025.
26. Vl. V. Voevodin, A. S. Antonov, D. A. Nikitenko, et al., "Supercomputer Lomonosov-2: Large Scale, Deep Monitoring and Fine Analytics for the User Community," Supercomput. Front. Innov. **6** (2), 4–11 (2019). doi 10.14529/jsfi190201.
27. System Architecture - TACC Frontera User Guide. https://frontera-portal.tacc.utexas.edu/user-guide/system/. Cited May 12, 2025.
28. GOST R ISO 22274–2016. Systems to manage terminology, knowledge and content. Concept-related aspects for developing and internationalizing classification systems (Standartinform Publ., Moscow, 2017) [in Russian].

2025, **26** (2), 192–207. doi 10.26089/NumMet.v26r214

29. GOST 8.417–2002. State system for ensuring the uniformity of measurements. Units of quantities (Standartinform, Moscow, 2019) [in Russian].

Received
February 28, 2025

Accepted for publication
May 10, 2025

Information about the authors

Alexander S. Antonov — Ph. D., Leading Researcher; Lomonosov Moscow State University, Research Computing Center, Leninskie Gory, 1, building 4, 119991, Moscow, Russia.

Anton A. Shcherbakov — master's student, Lomonosov Moscow State University, Research Computing Center, Leninskie Gory, 1, building 4, 119991, Moscow, Russia.