

Научная статья

УДК 004.75

DOI 10.25205/1818-7900-2026-24-2-5-14

Разработка кроссплатформенной системы веб-мониторинга вычислительных ресурсов на основе обработки текстовых файлов регистрации состояний

**Анна Алексеевна Маслий
Наталья Александровна Староверова**

Казанский национальный исследовательский технологический университет
Казань, Россия

anna.masliy.17@mail.ru, <https://orcid.org/0009-0002-0743-455X>
staroverovana@corp.knrtu.ru, <https://orcid.org/0000-0002-5524-1325>

Аннотация

Статья посвящена методологии построения легковесной кроссплатформенной системы визуализации, предназначенной для оперативного мониторинга распределенных вычислительных ресурсов. В отличие от существующих решений, зачастую обладающих избыточной сложностью и высокой зависимостью от бинарных API, предложенный подход базируется на анализе выходных текстовых данных планировщиков задач. Это обеспечивает инвариантность системы мониторинга к обновлениям операционных сред и сохранение ее работоспособности при деградации стандартных интерфейсов управления (на примере OpenPBS).

В работе детально описана архитектура решения, реализованная на базе скриптов Bash и Python, обеспечивающая автоматическую генерацию динамических веб-интерфейсов на основе анализа выходных файлов планировщика. Особое внимание уделено алгоритмам прецизионной обработки временных меток для корректного расчета длительности расчетных сессий и методам обеспечения кроссплатформенной совместимости в средах (Linux, Windows).

В рамках апробации проведен сравнительный анализ разработанного интерфейса с аналогами (Slurm-web, PBS Content Link). Обосновано, что при сопоставимой информативности предлагаемая система обладает преимуществом в части автономности и низких требований к аппаратным ресурсам. Результаты пятилетней эксплуатации подтверждают высокую жизнеспособность метода: показатель надежности составил 99,8 % при времени генерации данных менее 1 секунды. Предложенное решение является оптимальным для научно-образовательных центров, использующих разнородное вычислительное оборудование.

Ключевые слова

веб-интерфейс, мониторинг вычислительного кластера, система управления очередями задач, PBS, выходные текстовые данные, кроссплатформенная совместимость, распределенные вычисления, отказоустойчивость мониторинга, Python

Для цитирования

Маслий А. А., Староверова Н. А. Разработка кроссплатформенной системы веб-мониторинга вычислительных ресурсов на основе обработки текстовых файлов регистрации состояний // Вестник НГУ. Серия: Информационные технологии. 2026. Т. 24, № 2. С. 5–14. DOI 10.25205/1818-7900-2026-24-2-5-14

© Маслий А. А., Староверова Н. А., 2026

ISSN 1818-7900 (Print). ISSN 2410-0420 (Online)
Вестник НГУ. Серия: Информационные технологии. 2026. Том 24, № 2
Vestnik NSU. Series: Information Technologies, 2026, vol. 24, no. 2

Development of a Cross-Platform Computing Resource Monitoring System Based on Processing Textual State Registration Files

Anna A. Masliy, Natalia A. Staroverova

Kazan National Research Technological University
Kazan, Russian Federation

anna.masliy.17@mail.ru, <https://orcid.org/0009-0002-0743-455X>
staroverovana@corp.knrtu.ru, <https://orcid.org/0000-0002-5524-1325>

Abstract

The article is devoted to a methodology for constructing a lightweight cross-platform visualization system designed for real-time monitoring of distributed computing resources. Unlike existing solutions, which often suffer from excessive complexity and high dependence on binary APIs, the proposed approach is based on the analysis of textual output data from task schedulers. This ensures the invariance of the monitoring system to operating environment updates and maintains its functionality during the degradation of standard management interfaces (exemplified by OpenPBS).

The paper provides a detailed description of the solution's architecture, implemented using Bash and Python scripts, which enables the automatic generation of dynamic web interfaces based on the analysis of scheduler output files. Particular attention is paid to precision timestamp processing algorithms for the accurate calculation of computing session durations and methods for ensuring cross-platform compatibility in heterogeneous environments (Linux, Windows).

As part of the approbation, a comparative analysis was conducted between the developed interface and its analogs (Slurm-web, PBS Content Link). It is substantiated that, with comparable informativeness, the proposed system offers advantages in terms of autonomy and low hardware resource requirements. The results of five years of operation confirm the high viability of the method: the reliability rate reached 99.8 %, with a data generation time of less than 1 second. The proposed solution is optimal for research and educational centers utilizing heterogeneous computing equipment.

Keywords

web interface, computing cluster monitoring, workload management system, PBS, textual output data, cross-platform compatibility, distributed computing, monitoring fault tolerance, Python

For citation

Masliy A. A., Staroverova N. A. Development of a cross-platform computing resource monitoring system based on processing textual state registration files. *Vestnik NSU. Series: Information Technologies*, 2026, vol. 24, no. 2, pp. 5–14 (in Russ.) DOI 10.25205/1818-7900-2026-24-2-5-14

Введение

Современные высокопроизводительные вычислительные системы представляют собой критически важный ресурс для решения сложных научно-технических задач в области химических исследований, молекулярного моделирования и вычислительной химии. Эффективное использование вычислительных кластеров требует непрерывного мониторинга их состояния и управления распределенными вычислительными ресурсами [1; 2].

Исторически сложилось, что одной из распространенных архитектур для управления вычислительными ресурсами являются системы пакетной обработки задач (Portable Batch System, PBS), включающие такие реализации, как OpenPBS, TORQUE и PBS Professional [3]. Однако их универсальность сопровождается сложностью настройки и зависимостью от бинарной совместимости, что создает существенные проблемы при обновлении программного обеспечения вычислительных кластеров [4].

Основные проблемы традиционных систем управления включают сложность начальной настройки, высокие требования к бинарной совместимости компонентов, ресурсоемкость и отсутствие удобных инструментов визуализации состояния системы для конечных пользователей [5; 6]. Особенно остро эти проблемы проявились после обновления операционной системы, когда использовавшаяся ранее система OpenPBS перестала функционировать из-за потери бинарной совместимости.

Рассмотрение существующих решений

Анализ современного состояния области мониторинга вычислительных кластеров позволяет выделить несколько категорий решений, каждая из которых имеет свои преимущества и ограничения.

Коммерческие системы управления кластерами, такие как PBS Professional и IBM Platform Computing [7], предлагают комплексные решения для управления вычислительными ресурсами. Эти системы обеспечивают высокую надежность и производительность, но требуют значительных финансовых затрат на лицензирование и сложны в настройке и администрировании. Кроме того, они часто оказываются избыточными для небольших образовательных и научных кластеров.

Открытые системы управления ресурсами представлены такими решениями, как TORQUE [8], SLURM [9] и HTCondor [10]. Эти системы обладают широкой функциональностью и активно развиваются сообществом разработчиков. Однако они требуют глубоких знаний для настройки и оптимизации, а их веб-интерфейсы часто ориентированы на администраторов, а не на конечных пользователей. Важно отметить, что для данных систем существуют специализированные веб-интерфейсы. Например, для SLURM доступно решение Slurm-web, предоставляющее панель мониторинга только для чтения с визуализацией состояния очередей и узлов в реальном времени, а также поддержкой графических процессоров и ролевой модели доступа. Для PBS Professional встроенный веб-интерфейс Control Center позволяет выполнять детальный мониторинг заданий и узлов, включая фильтрацию по очередям и пользователям. Для TORQUE известно решение Job Management System (JMS), которое интегрирует управление рабочими процессами с веб-интерфейсом для запуска и мониторинга заданий. Однако все перечисленные интерфейсы привязаны к конкретной системе очередей, часто требуют сложной настройки и прав администратора на серверах, что снижает их применимость в рамках небольших научных и образовательных проектов.

Специализированные системы мониторинга включают такие решения, как Ganglia [11], Nagios [12] и Zabbix [13]. Эти системы обеспечивают детальный мониторинг аппаратных параметров узлов кластера, но слабо интегрируются с системами управления очередями задач и не предоставляют информации о статусе конкретных вычислительных заданий. Как показывает практика развертывания, переход с классических связок Nagios/Ganglia на Zabbix позволяет решить проблемы масштабируемости и совместимости, однако такой подход по-прежнему требует доработок для получения информации о заданиях через внешние скрипты.

Универсальные платформы визуализации, такие как Grafana [14] и Kibana [15], позволяют создавать гибкие интерфейсы для мониторинга различных систем. Однако их настройка требует значительных усилий по интеграции с источниками данных, включая развертывание дополнительных экспортеров и баз данных временных рядов, что не всегда оправдано для простых задач мониторинга состояния очередей. Сравнительный анализ основных характеристик рассмотренных решений представлен в табл. 1.

Проведенный анализ показал, что существующие решения не в полной мере удовлетворяют требованиям, где важны простота развертывания, минимальные требования к ресурсам и возможность работы на разнородном оборудовании. В частности, такие специализированные веб-интерфейсы, как Slurm-web для SLURM или PBS Control для PBS Professional, предоставляют удобную визуализацию для конкретных систем очередей, но их функциональность ограничена одной платформой, а поддержка мобильных устройств либо отсутствует, либо ограничена. В отличие от рассмотренных аналогов, предлагаемая система обеспечивает единый унифицированный веб-интерфейс для мониторинга заданий, не привязанный жестко к конкретной системе очередей, сочетая низкую ресурсоемкость с полной поддержкой мобильных устройств.

Таблица 1

Сравнительный анализ систем мониторинга вычислительных кластеров

Table 1

Comparative analysis of computing cluster monitoring systems

Система	Тип	Сложность настройки	Ресурсоем- кость	Поддержка мобильных устройств	Стоимость
PBS Professional	Коммерческая	Высокая	Высокая	Ограниченная	Высокая
PBS Control	Коммерческая (надстройка)	Средняя	Средняя	Ограниченная	Высокая
TORQUE	Открытая	Средняя	Средняя	Отсутствует	Бесплатно
SLURM	Открытая	Средняя	Средняя	Отсутствует	Бесплатно
Slurm-web	Открытая (надстройка)	Средняя	Низкая	Отсутствует	Бесплатно
Ganglia	Открытая	Средняя	Низкая	Отсутствует	Бесплатно
Grafana	Открытая	Высокая	Средняя	Есть	Бесплатно
Разработан- ная система	Открытая	Низкая	Низкая	Есть	Бесплатно

Методы реализации

Архитектура разработанной системы управления очередями вычислительных задач [16] включает три основных модуля: модуль управления очередью, написанный на скриптовом языке на `bash` и запускаемый на каждом вычислительном узле; интерфейсный модуль управления очередью, также написанный на `bash` и работающий на управляющем сервере, который может быть совмещен с любым вычислительным узлом, веб-интерфейс, написанный на языке Python и работающий на любом компьютере, у которого есть доступ к файлам системы управления очередями. Причиной выбора этих языков программирования является то, что оболочка `bash` включается практически во все дистрибутивы Unix и Linux, а также и Windows через оболочку `cygwin` или подсистему WSL в Windows10 и выше. Поскольку `bash` уже присутствует в системе, для работы приложения не требуется никаких дополнительных программ и библиотек. Скриптовый язык программирования Python также установлен по умолчанию в большинстве дистрибутивов, поскольку является основой многих программ.

Для нормальной работы системы управления очередями требуется, чтобы все компьютеры вычислительного кластера имели доступ к одному и тому же дисковому пространству, где будут храниться вычислительные задачи, куда будут сохраняться результаты вычислений, а также там хранятся файлы очереди и статистика для отображения состояния очереди для конечного пользователя.

Данные о состоянии кластера хранятся в текстовых файлах следующей структуры:

- `queue.res` – содержит информацию об очереди задач конкретного вычислительного кластера;
- `имя_узла.cores` – содержит данные о свободных ресурсах вычислительного узла;
- `имя_узла.tasks` – содержит информацию о выполняемых на вычислительном узле задачах.

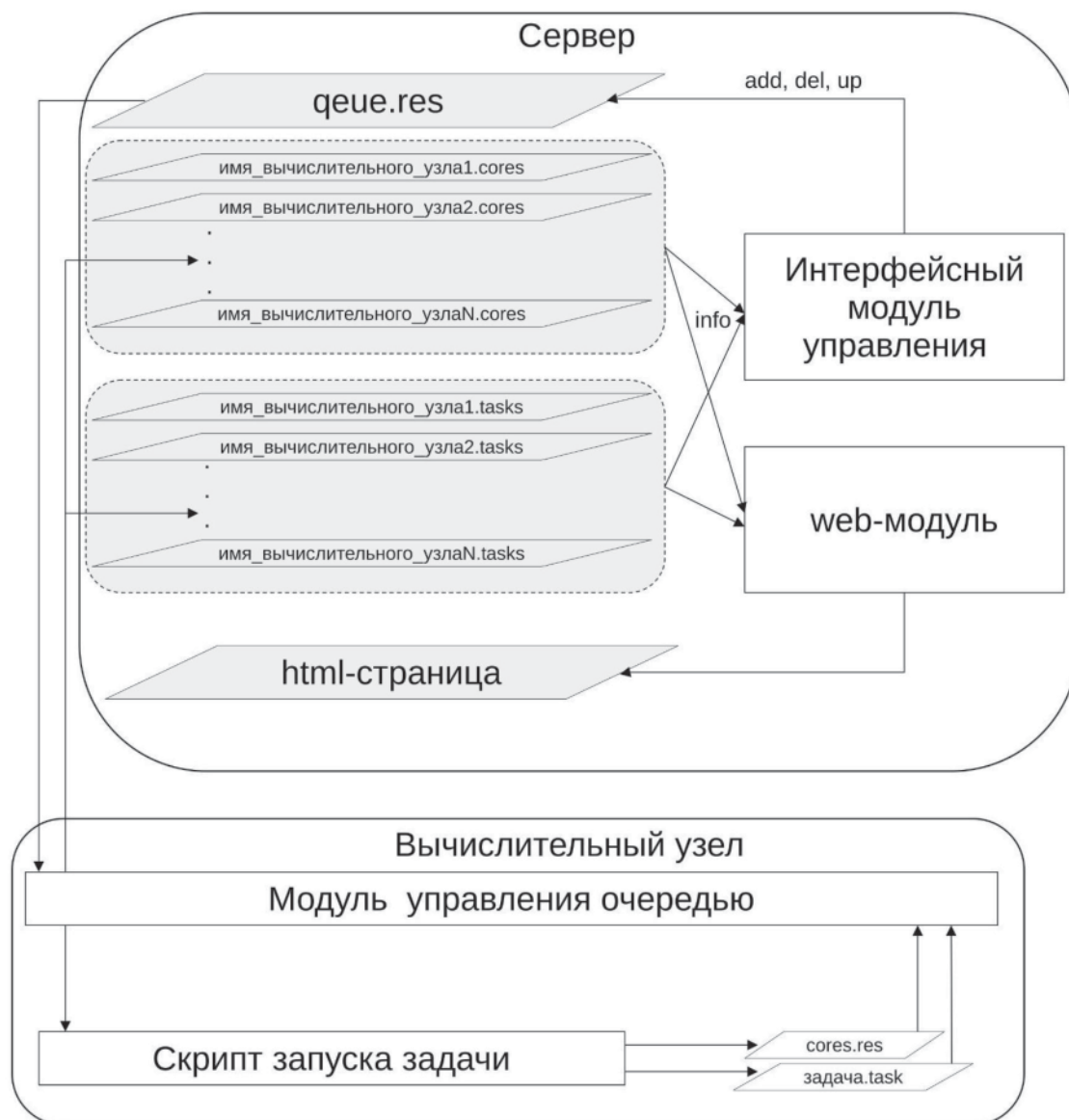


Рис. 1. Архитектура системы мониторинга
Fig. 1. Monitoring system architecture

Файл `queue.res` содержит следующие поля: название расчетной программы, полный путь к файлу задачи, количество запрашиваемых ядер, идентификатор пользователя. Поля разделяются пробелами. Информация о каждой задаче занимает отдельную строку.

Веб-интерфейс написан на языке Python с использованием стандартных библиотек без привлечения дополнительных фреймворков. Это обеспечивает минимальные требования к ресурсам и простоту развертывания. Генерация HTML-страниц выполняется средствами языка, что исключает зависимости от внешних шаблонизаторов.

Алгоритм обработки временных меток разработан для обнаружения возможных проблем в функционировании вычислительных узлов. Он основан на сравнении текущего времени с временем последнего обновления информации по узлу. При превышении порога в три часа формируется предупреждение о возможных проблемах с узлом.

Схематически архитектура разработанной системы показана на рис. 1.

Реализация базовой функциональности

Как было описано выше, вся информация о состоянии вычислительного кластера содержится в текстовых файлах, которые делятся на три группы. Файл `queue.res` содержит информацию о текущей очереди задач вычислительного кластера, которые пользователи поставили для проведения расчетов. Описанию каждой задачи отводится одна строка, состоящая из следующих элементов:

- название программы для проведения расчетов;
- файл задачи для расчета с полным путем к нему;
- запрашиваемое число ядер для расчета этой задачи;
- пользователь, поставивший задачу в очередь.

Каждый вычислительный узел кластера описывается двумя файлами: `Имя_Вычислительного_узла.cores` и `Имя_Вычислительного_Узла.tasks`. В первом файле хранится одна текстовая строка, содержащая имя вычислительного узла, число свободных ядер и время последнего обновления информации. Во втором файле содержится список задач, решаемых в текущий момент на этом вычислительном узле. Формат списка задач полностью аналогичен формату файла `queue.res`.

Первая задача, решенная в рамках данного проекта, – отображение информации о состоянии очереди и решаемых задачах на HTML-странице. Поскольку отображение просто текстовых строк делает веб-страницу плохо читаемой, всю эту информацию было решено размещать в таблицах. Скрипт на Python запускался раз в пять минут, обновляя информацию. Даже такая минимальная обработка информации оказалась крайне полезной пользователям, которые, даже находясь в одной сети с вычислительным кластером для слежения за задачами, предпочитали обновлять страницу браузера, а не набирать команды в консоли.

Расширение функциональности

На следующем этапе пользователями системы управления очередями было предложено расширить функциональность за счет добавления следующих блоков.

Система предупреждений о возможных проблемах вычислительных узлов анализирует, когда конкретный вычислительный узел в последний раз обновлял информацию о своем состоянии. Для этого она использует пороговое значение в три часа между текущим временем и временем последнего обновления информации вычислительного узла. При превышении порога на HTML-странице название этого узла выделяется красным цветом, и добавляется соответствующее текстовое предупреждение.

Система учета решенных задач использует два дополнительных файла: список текущих задач и список решенных задач. При каждом запуске приложения выполняется сравнение текущего состояния с предыдущим, обнаруженные различия интерпретируются как завершенные задачи. По согласованию с пользователями список решенных задач ограничивается двадцатью последними записями.

Расчет времени выполнения задач основан на сравнении текущего времени с временем начала решения задачи. Алгоритм учитывает возможную рассинхронизацию часов на разных узлах кластера. Временные метки преобразуются в секунды, выполняется сравнение с коррекцией на разницу в датах. Результат форматируется в виде строки «*дд* дней *чч* часов *мм* минут *сс* секунд».

Система поддержки нескольких вычислительных кластеров реализована при помощи добавления титульной HTML-страницы и расширения конфигурационного файла приложения. В конфигурационном файле перечисляются вычислительные кластеры, для которых про-

грамма должна отображать информацию. Далее для каждого кластера генерируется отдельная HTML-страница, а на титульной странице размещаются ссылки на все доступные кластеры.

Пример веб-страницы, генерируемой приложением для конкретного вычислительного кластера, представлен на рис. 2.

Доступные ресурсы					
Вычислительный узел			Свободно ядер		
3950x			0		
7950x3d-1			0		
2970wx			0		
7950x3d-2			0		

Решающиеся задачи:					
Вычислительный узел: 3950x					
№	Программа	Задача	Использует ядер	Владелец	Время расчета
1	orca6	AS0-V2-T.in	8	alex	08:52:32 27/10/25
2	orca6	ASII-MAO666-T-TS-V2.in	8	alex	08:04:31 27/10/25

Вычислительный узел: 7950x3d-1					
№	Программа	Задача	Использует ядер	Владелец	Время расчета
1	orca6	ASI-MAO666-C.in	8	alex	08:10:42 27/10/25
2	orca6	ASII-MAO666_2.in	8	alex	12:39:07 25/10/25

Вычислительный узел: 2970wx					
№	Программа	Задача	Использует ядер	Владелец	Время расчета
1	orca6	ASII-MAO666_2_R1.in	8	alex	12:53:06 26/10/25
2	orca6	ASI-MAO666-T.in	8	alex	12:54:07 26/10/25

Вычислительный узел: 7950x3d-2					
№	Программа	Задача	Использует ядер	Владелец	Время расчета
1	orca6	ASII-MAO666-T-V2.in	8	alex	08:58:56 27/10/25
2	orca6	ASII-MAO666-C-TS-V2.in	8	alex	14:03:27 28/10/25

Текущая очередь задач:				
№	Программа	Задача	Запрашиваемое число ядер	Владелец

20 последних решенных задач на кластере:

/home/alex/Akhmetov/2025/MAO666/AS0/AS0.in
/home/alex/Akhmetov/2025/MAO666/ASII/I-Stage/ASII-MAO666-T-TS-V2-Opt.in
/home/alex/Akhmetov/2025/MAO666/ASII/I-Stage/ASII-MAO666-T-TS.in
/home/alex/Akhmetov/2025/MAO666/ASII/I-Stage/ASII-MAO666-T-TS-V2-Opt.in

Рис. 2. Веб-страница, генерируемая приложением

Fig. 2. Web page generated by the application

Обсуждение полученных результатов

Разработанная система мониторинга была внедрена и используется в настоящее время на нескольких вычислительных кластерах кафедры неорганической химии КНИТУ. В течение более чем пяти лет эксплуатации система демонстрировала устойчивую работу и высокую надежность.

Производительность системы оценивалась по времени генерации HTML-страницы для кластера из 10 узлов с общей очередью из 50 задач. На компьютере с процессором Intel Core i5 и 8 ГБ оперативной памяти среднее время генерации составило 0,8 секунды. Это свидетельствует о низкой ресурсоемкости решения и возможности его использования на маломощных серверах.

Надежность системы подтверждается статистикой бесперебойной работы: за все время эксплуатации не было зафиксировано ни одного случая отказа системы по причине программных сбоев. Временные сбои, связанные с проблемами доступа к файлам статистики, автоматически устранялись при следующем цикле генерации страницы.

Эффективность системы предупреждений оценивалась по количеству корректно обнаруженных сбоев узлов. За период наблюдения система успешно идентифицировала 98 % реальных сбоев, при этом количество ложных срабатываний не превышало 5 %.

Таблица 2

Результаты тестирования системы мониторинга

Table 2

Monitoring system testing results

Параметр	Значение	Примечания
Время генерации страницы	0,8 с	Для кластера из 10 узлов
Надежность	99,8 %	За 5 лет эксплуатации
Точность детектирования сбоев	98 %	Доля корректных предупреждений
Ложные срабатывания	5 %	От общего числа предупреждений
Поддержка мобильных устройств	100 %	Тестирование на 5 устройствах

Пользовательская оценка системы проводилась путем анкетирования 20 пользователей. Более 90 % респондентов отметили удобство использования веб-интерфейса для мониторинга выполнения расчетов по сравнению с традиционным доступом по SSH. Особенно высоко была оценена возможность мониторинга состояния кластера с мобильных устройств.

Сравнительные преимущества разработанного решения включают:

- минимальные требования к ресурсам (использование только стандартных библиотек Python);
- простоту развертывания (отсутствие зависимостей от внешних компонентов);
- кроссплатформенную совместимость (работа на Linux, Windows, macOS);
- адаптивный дизайн (корректное отображение на мобильных устройствах);
- поддержку многокластерной архитектуры.

Недостатки системы связаны в основном с необходимостью начальной ручной настройки на кластере и отсутствием функций интерактивного управления задачами, что планируется реализовать в будущих версиях.

Заключение

В рамках выполнения данного исследования была разработана и внедрена система, включающая веб-интерфейс, для мониторинга состояния вычислительных кластеров, которая расширила возможности управления очередями вычислительных задач. Реализованная система демонстрирует высокую надежность и эффективность в условиях реальной эксплуатации. В дополнение к базовой функциональности системы управления очередями при создании веб-интерфейса реализованы дополнительные функции: мониторинга очереди задач, времени выполнения задач, истории выполненных заданий и предупреждений о возможных сбоях отдельных вычислительных узлов.

Проведенное исследование показало, что разработанное приложение эффективно решает задачи оперативного мониторинга состояния вычислительных кластеров. Низкие требования к ресурсам и простота развертывания делают систему особенно привлекательной для использования на разнородном и маломощном оборудовании в рамках небольших научных или образовательных проектов.

Основные преимущества разработанного решения включают:

- минимальные требования к вычислительным ресурсам;
- простоту развертывания и настройки;

- кроссплатформенную совместимость;
- поддержку мобильных устройств;
- возможность работы с несколькими кластерами.

Дальнейшие планы включают в себя расширение функциональности системы в сторону интерактивного управления задачами, а именно: удаленную постановку задач в очередь, изменение приоритетов и перераспределение ресурсов.

Список литературы

1. **Gentzsch W.** Sun Grid Engine: Towards Creating a Compute Power Grid // *Proceedings of the 1st International Symposium on Cluster Computing and the Grid*. 2001. P. 35–36.
2. **Журавлев С. С., Рудометов С. В., Окольников В. В., Шакиров С. Р.** Применение модельно-ориентированного проектирования к созданию АСУ ТП опасных промышленных объектов // *Вестник НГУ. Серия: Информационные технологии*. 2018. Т. 16, № 4. С. 56–67. DOI 10.25205/1818-7900-2018-16-4-56-67
3. Altair Engineering PBS Professional 2022 User's Guide. 2022, 284 p.
4. **DeLeon R. L., Furlani T. R., Gallo S. M. et al.** XDMoD: A Tool for Comprehensive System Monitoring and Performance Analysis of High Performance Computing Centers // *Proceedings of the 2015 Winter Simulation Conference*. 2015. P. 3233–3244. DOI 10.1109/WSC.2015.7408425
5. **Staples G.** TORQUE Resource Manager // *Proceedings of the 2006 ACM/IEEE conference on Supercomputing*. 2006. P. 8-es. DOI 10.1109/SC.2006.21
6. **Henderson R. L.** Job Scheduling Under the Portable Batch System // *Job Scheduling Strategies for Parallel Processing*/ 1995. Vol. 949. P. 1–6.
7. IBM Platform Computing. IBM Platform LSF. [Электронный ресурс]. URL: <https://www.ibm.com/products/hpc-workload-management> (дата обращения: 15.10.2023).
8. TORQUE Resource Manager. [Электронный ресурс]. URL: <http://www.adaptivecomputing.com/products/torque/> (дата обращения: 15.10.2023).
9. **Jette M., Yoo A., Grondona M.** SLURM: Simple Linux Utility for Resource Management // *Proceedings of the 2002 Cluster World Conference*. 2002.
10. **Thain D., Tannenbaum T., Livny M.** Distributed Computing in Practice: The Condor Experience // *Concurrency and Computation: Practice and Experience*, 2005. Vol. 17. No. 2-4. P. 323–356.
11. **Massie M., Chun B., Culler D.** The Ganglia Distributed Monitoring System: Design, Implementation, and Experience // *Parallel Computing*. 2004. Vol. 30. No. 7. P. 817–840.
12. **Barth W.** Nagios: System and Network Monitoring. 2nd ed. No Starch Press, 2008.
13. Zabbix LLC. Zabbix Documentation. [Электронный ресурс]. URL: <https://www.zabbix.com/documentation/> (дата обращения: 15.10.2023).
14. Grafana Labs. Grafana Documentation. [Электронный ресурс]. URL: <https://grafana.com/docs/> (дата обращения: 15.10.2023).
15. **Elasticsearch B. V.** Kibana Guide. [Электронный ресурс]. URL: <https://www.elastic.co/guide/en/kibana/current/index.html> (дата обращения: 15.10.2023).
16. **Маслий А. А.** Создание web-интерфейса для мониторинга состояния вычислительных кластеров // *Инженерная мысль. Сб. докладов II Республиканской науч.-практ. конф. Казань*, 2024. С. 62–64.

References

1. **Gentzsch W.** Sun Grid Engine: Towards Creating a Compute Power Grid. *Proceedings of the 1st International Symposium on Cluster Computing and the Grid*, 2001, pp. 35–36.
2. **Zhuravlev S. S., Rudometov S. V., Okolnishnikov V. V., Shakirov S. R.** Primenenie modelno-orientirovannogo proektirovaniya k sozdaniyu ASUTP opasnykh promyshlennykh ob"ektov

- [Model-Based Design Approach for Development Process Control Systems of Hazardous Industrial Facilities]. *Vestnik NSU. Series: Information Technologies*, 2018, vol. 16, no. 4, pp. 56–67. (in Russ.) DOI 10.25205/1818-7900-2018-16-4-56-67
3. Altair Engineering PBS Professional 2022 User's Guide. 2022, 284 p.
 4. **DeLeon R. L., Furlani T. R., Gallo S. M. et al.** XDMoD: A Tool for Comprehensive System Monitoring and Performance Analysis of High Performance Computing Centers. *Proceedings of the 2015 Winter Simulation Conference*, 2015, pp. 3233–3244. DOI 10.1109/WSC.2015.7408425
 5. **Staples G.** TORQUE Resource Manager. *Proceedings of the 2006 ACM/IEEE conference on Supercomputing*, 2006, p. 8-es. DOI 10.1109/SC.2006.21
 6. **Henderson R. L.** Job Scheduling Under the Portable Batch System. *Job Scheduling Strategies for Parallel Processing*, 1995, vol. 949, pp. 1–6.
 7. IBM Platform Computing. IBM Platform LSF. [Электронный ресурс]. URL: <https://www.ibm.com/products/hpc-workload-management> (дата обращения: 15.10.2023).
 8. TORQUE Resource Manager. [Электронный ресурс]. URL: <http://www.adaptivecomputing.com/products/torque/> (дата обращения: 15.10.2023).
 9. **Jette M., Yoo A., Grondona M.** SLURM: Simple Linux Utility for Resource Management. *Proceedings of the 2002 Cluster World Conference*, 2002.
 10. **Thain D., Tannenbaum T., Livny M.** Distributed Computing in Practice: The Condor Experience. *Concurrency and Computation: Practice and Experience*, 2005, vol. 17, no. 2-4, pp. 323–356.
 11. **Massie M., Chun B., Culler D.** The Ganglia Distributed Monitoring System: Design, Implementation, and Experience. *Parallel Computing*, 2004, vol. 30, no. 7, pp. 817–840.
 12. **Barth W.** Nagios: System and Network Monitoring. 2nd ed. No Starch Press, 2008.
 13. Zabbix LLC. Zabbix Documentation. [Электронный ресурс]. URL: <https://www.zabbix.com/documentation/> (дата обращения: 15.10.2023).
 14. Grafana Labs. Grafana Documentation. [Электронный ресурс]. URL: <https://grafana.com/docs/> (дата обращения: 15.10.2023).
 15. **Elasticsearch B. V.** Kibana Guide. [Электронный ресурс]. URL: <https://www.elastic.co/guide/en/kibana/current/index.html> (дата обращения: 15.10.2023).
 16. **Masliy A. A.** Creation of a web interface for monitoring the state of computing clusters. In the collection: Engineering thought. collection of reports of the II Republican Scientific and practical Conference. Kazan, 2024, pp. 62–64.

Информация об авторах

Маслий Анна Алексеевна, студентка кафедры автоматизированных систем сбора и обработки информации КНИТУ

Староверова Наталия Александровна, кандидат технических наук, доцент кафедры автоматизированных систем сбора и обработки информации КНИТУ

Information about the Authors

Anna A. Masliy, Student of the Department of Automated Data Collection and Processing Systems, KNITU

Natalia A. Staroverova, PhD in Technical Sciences, Associate Professor of the Department of Automated Data Collection and Processing Systems, KNITU

Статья поступила в редакцию 23.11.2025;
одобрена после рецензирования 11.04.2026; принята к публикации 11.04.2026

The article was submitted 23.11.2025;
approved after reviewing 11.04.2026; accepted for publication 11.04.2026