УДК 004.75 *DOI 10.55421/3034-4689_2026_29_6_108*

А. А. Маслий, Н. А. Староверова

СОЗДАНИЕ WEB-ИНТЕРФЕЙСА ДЛЯ МОНИТОРИНГА СОСТОЯНИЯ ВЫЧИСЛИТЕЛЬНЫХ КЛАСТЕРОВ

Ключевые слова: web-интерфейс, мониторинг, вычислительный кластер, система управления очередями, PBS, Python, HTML, распределенные вычисления, высокопроизводительные вычисления, энергоэффективность.

В работе рассматривается разработка легковесного веб-интерфейса для мониторинга состояния вычислительных кластеров, функционирующих под управлением специализированной системы управления очередями задач. Актуальность решения обусловлена необходимостью простого и эффективного инструмента для оперативного контроля за распределенными вычислительными ресурсами в научно-образовательной среде, где зачастую используются разнородные и маломощные системы. Представлена архитектура решения, основанная на анализе текстовых файлов статистики и их автоматической трансформации в HTML-страницы. Система построена на использовании скриптов Bash и Python, что обеспечивает минимальные зависимости и простоту развертывания. Описаны основные функциональные возможности системы, включая отображение текущей очереди задач, мониторинг загрузки вычислительных узлов (процессоры, память), детектирование сбоев на основе анализа временных меток и формирование истории выполненных заданий. Особое внимание уделено методам обработки временных меток для обеспечения корректного расчета времени выполнения задач и алгоритмам обеспечения кроссплатформенной совместимости (Linux, Windows). Разработанное решение прошло успешную апробацию в реальных эксплуатационных условиях и демонстрирует устойчивую работу в течение пяти лет на нескольких вычислительных кластерах. Приведены результаты тестирования системы, подтверждающие ее высокую эффективность: время генерации страницы для кластера из 10 узлов составляет менее 1 секунды, точность детектирования сбоев достигает 98%, а надежность за период эксплуатации оценена в 99,8%. Ключевыми преимуществами системы являются низкие требования к вычислительным ресурсам, простота развертывания, адаптивный дизайн для мобильных устройств и поддержка многокластерной архитектуры, что делает ее особенно привлекательной для образовательных учреждений и небольших научных групп.

A. A. Masliy, N. A. Staroverova

CREATING A WEB INTERFACE FOR MONITORING THE STATUS OF COMPUTING CLUSTERS

Key words: web interface, monitoring, computing cluster, job scheduling system, PBS, Python, HTML, distributed computing, high-performance computing, energy efficiency.

The paper considers the development of a lightweight web interface for monitoring the status of computing clusters operating under the control of a specialized task queue management system. The relevance of the solution is due to the need for a simple and effective tool for operational control of distributed computing resources in a scientific and educational environment where heterogeneous and low-power systems are often used. The architecture of the solution based on the analysis of text files of statistics and their automatic transformation into HTML pages is presented. The system is based on the use of Bash and Python scripts, which ensures minimal dependencies and ease of deployment. The basic functionality of the system is described, including displaying the current task queue, monitoring the load of computing nodes (processors, memory), detecting failures based on timestamp analysis, and creating a history of completed tasks. Special attention is paid to timestamp processing methods to ensure correct calculation of task execution time and algorithms for cross-platform compatibility (Linux, Windows). The developed solution has been successfully tested in real-world operating conditions and has demonstrated stable performance over five years on several computing clusters. The results of testing the system are presented, confirming its high efficiency: the page generation time for a cluster of 10 nodes is less than 1 second, the error detection accuracy reaches 98%, and reliability over the period of operation is estimated at 99.8%. The key advantages of the system are low demands on computing resources, ease of deployment, adaptive design for mobile devices and support for multicluster architecture, which makes it especially attractive for educational institutions and small research groups.

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 создаёт 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. Краткий сравнительный анализ приведен в таблице 1.

Таблица 1 - Сравнительный анализ систем мониторинга вычислительных кластеров

Table 1 - Comparative Analysis of Computing Cluster Monitoring Systems

Система	Тип	Сложность настройки	Ресурсоемкость	Поддержка мобильных устройств	Стоимость
PBS Professional	Коммерческая	Высокая	Высокая	Ограниченная	Высокая
TORQUE	Открытая	Средняя	Средняя	Отсутствует	Бесплатно
SLURM	Открытая	Средняя	Средняя	Отсутствует	Бесплатно
Ganglia	Открытая	Средняя	Низкая	Отсутствует	Бесплатно
Grafana	Открытая	Высокая	Средняя	Есть	Бесплатно
Разработанная система	Открытая	Низкая	Низкая	Есть	Бесплатно

Коммерческие системы управления кластерами, такие как PBS Professional [3] 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.

Специализированные системы мониторинга включают такие решения, как Ganglia [11], Nagios [12] и Zabbix [13]. Эти системы обеспечивают детальный мониторинг hardware-параметров узлов кластера, но слабо интегрируются с системами управления очередями задач и не предоставляют информации о статусе конкретных вычислительных заданий.

Универсальные платформы визуализации, такие как Grafana [14] и Kibana [15], позволяют создавать sophisticated-интерфейсы для мониторинга различных систем. Однако их настройка требует значительных усилий по интеграции с источниками данных и не всегда оправдана для простых задач мониторинга состояния очередей.

Проведенный анализ показал, что существующие решения не в полной мере удовлетворяют требованиям образовательных учреждений, где важны простота развертывания, минимальные требования к ресурсам и возможность работы на разнородном оборудовании.

Постановка задачи

Целью работы являлась разработка веб-интерфейса для мониторинга состояния вычислительных кластеров, обеспечивающего:

- отображение текущей очереди задач;
- мониторинг загрузки вычислительных узлов;
- список выполняемых и завершённых задач;
- предупреждения о возможных сбоях в работе кластера;
- поддержку многокластерной архитектуры.

Для достижения поставленной цели необходимо было решить следующие задачи:

- анализ существующих систем управления очередями;
- разработка форматов хранения статистической информации;
- создание механизма генерации HTML-страниц;
- реализация алгоритмов обработки временных меток;
- разработка системы предупреждений о проблемах узлов.

Дополнительными требованиями к системе были: минимальные требования к вычислительным ресурсам, простота развертывания и кроссплатформенная совместимость. Необходимость написания веб-интерфейса обуславливается тем, что при использовании консольного приложения необходимо подключаться к серверу по протоколу SSH, что крайне неудобно при использовании мобильных устройств.

Методы реализации

Архитектура разработанной системы [16] включает три основных модуля: модуль управления очередью на bash, запускаемую на каждом вычислительном узле, интерфейсный модуль управления очередью на bash, работающий на сервере, веб-интерфейс на Python, работающий на сервере (рис. 1). Причиной такого выбора 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.

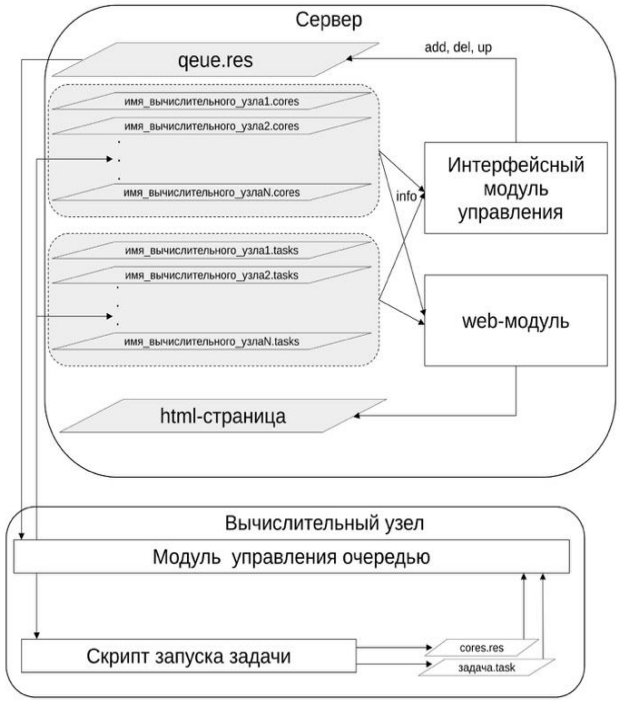


Рис. 1 - Архитектура системы мониторинга

Fig. 1 - Monitoring System Architecture

Для функционирования системы требуется чтобы все компьютеры вычислительного 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.

Для хранения данных о состоянии кластера используются текстовые файлы следующей структуры:

• *queue.res* — информация об очередях задач;

• *имя_узла.cores* — данные о свободных ресурсах узла;

• *имя_узла.tasks* — список выполняемых задач.

Формат файла queue.res включает следующие поля: название расчётной программы, полный путь к файлу задачи, количество запрашиваемых ядер, идентификатор пользователя. Каждая задача занимает отдельную строку.

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

Для обработки временных меток разработан алгоритм сравнения текущего времени с временем последнего обновления информации по узлу. При превышении порога в 3 часа формируется предупреждение о возможных проблемах с узлом.

Реализация базовой функциональности

После знакомства с системой управления задачами выяснилось, что вся информация о состоянии вычислительного кластера содержится в текстовых файлах, которые делятся на три группы. Файл queue.res содержит информацию о текущей очереди задач. Каждой задаче 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 в файле. Во втором содержится список задач, решаемых в текущий момент на этом вычислительном узле.

Первой задачей, которую необходимо было решить в рамках данного проекта, являлась задача отображения информации о состоянии очереди и решающихся задачах на html-странице. Поскольку отображение просто текстовых строк сделает веб-страницу плохо читаемой, всю эту информацию было решено размещать в таблицах.

Расширение функциональности

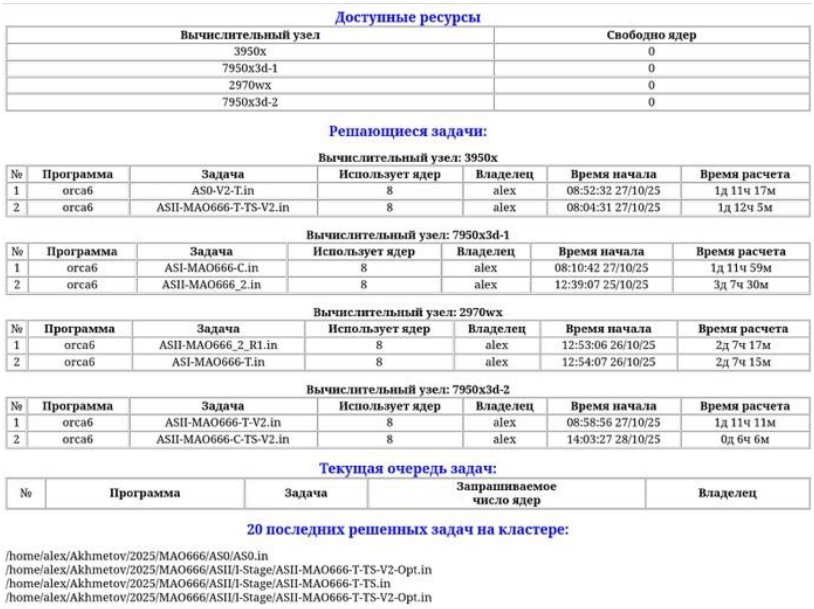
Система предупреждений о проблемах узлов использует пороговое значение в 3 часа между текущим временем и временем последнего обновления информации узла. При превышении порога название узла выделяется красным цветом, и добавляется соответствующее текстовое предупреждение.

Механизм учёта решённых задач использует два дополнительных файла: список текущих задач и список решённых задач. При каждом запуске приложения выполняется сравнение текущего состояния с предыдущим, обнаруженные различия интерпретируются как завершённые задачи. Список решённых задач ограничивается 20 последними записями.

Расчёт 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».

Поддержка многокластерности реализована через модификацию конфигурационного файла. Для каждого кластера генерируется отдельная HTML-страница, а на главной странице размещаются ссылки на все доступные кластеры.

Веб-страница, гернерируемая итоговой версией программа показана на рисунке 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 начала	Время расчета
1	orca6	AS0-V2-T.in	8	alex	08:52:32 27/10/25	1д 11ч 17м
2	orca6	ASII-MAO666-T-TS-V2.in	8	alex	08:04:31 27/10/25	1д 12ч 5м

Вычислительный узел: 7950x3d-1

№	Программа	Задача	Использует ядер	Владелец	Время начала	Время расчета
1	orca6	ASI-MAO666-C.in	8	alex	08:10:42 27/10/25	1д 11ч 59м
2	orca6	ASII-MAO666_2.in	8	alex	12:39:07 25/10/25	3д 7ч 30м

Вычислительный узел: 2970wx

№	Программа	Задача	Использует ядер	Владелец	Время начала	Время расчета
1	orca6	ASII-MAO666_2_R1.in	8	alex	12:53:06 26/10/25	2д 7ч 17м
2	orca6	ASI-MAO666-T.in	8	alex	12:54:07 26/10/25	2д 7ч 15м

Вычислительный узел: 7950x3d-2

№	Программа	Задача	Использует ядер	Владелец	Время начала	Время расчета
1	orca6	ASII-MAO666-T-V2.in	8	alex	08:58:56 27/10/25	1д 11ч 11м
2	orca6	ASII-MAO666-C-TS-V2.in	8	alex	14:03:27 28/10/25	0д 6ч 6м

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 прошла апробацию в реальных эксплуатационных условиях на нескольких вычислительных кластерах кафедры неорганической химии КНИТУ. В течение более чем пяти лет эксплуатации система демонстрировала устойчивую работу и высокую надежность. Результаты тестирования представлены в таблице 2.

Таблица 2 - Результаты тестирования системы мониторинга

Table 2 - Monitoring System Test 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

Производительность системы оценивалась по времени генерации HTML-страницы для кластера из 10 узлов с общей очередью из 50 задач. Среднее время генерации составило 0.8 секунды на компьютере с процессором Intel Core i5 и 8 ГБ оперативной памяти. Это свидетельствует о низкой ресурсоемкости решения и возможности его использования на маломощных серверах.

Надежность системы подтверждается статистикой бесперебойной работы: за пять лет эксплуатации не было зафиксировано ни одного случая полного отказа системы. 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%.

Заключение

В ходе работы был разработан веб-интерфейс для мониторинга состояния вычислительных кластеров, который успешно эксплуатируется в течение пяти лет. Система демонстрирует высокую надежность и эффективность в условиях реальной эксплуатации. Реализованы функции мониторинга очереди задач, загрузки узлов, истории выполненных заданий и предупреждений о сбоях.

Проведенное исследование показало, что разработанное решение эффективно решает задачи оперативного мониторинга состояния вычислительных кластеров в условиях образовательных учреждений. Низкие требования к ресурсам и простота развертывания делают систему особенно привлекательный для использования на разнородном и маломощном оборудовании.

Основные преимущества разработанного решения включают:

- минимальные требования к вычислительным ресурсам;
- простоту развертывания и настройки;
- кроссплатформенную совместимость;
- поддержку мобильных устройств;
- возможность работы с несколькими кластерами.

В дальнейшем планируется расширение функциональности системы в сторону интерактивного управления задачами, включая удаленную постановку задач в очередь, изменение приоритетов и перераспределение ресурсов.

Литература

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. Журавлев С. С., Рудометов С. В., Окольнишников 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, 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. Маслий А.А. Создание web-интерфейса для мониторинга состояния вычислительных кластеров. В сборнике: Инженерная мысль. сборник докладов II Республиканской научно-практической конференции. 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. Application of Model-Driven Design to the Development of Automated Process Control Systems for Hazardous Industrial Facilities. NSU Bulletin. Series: Information Technologies, 2018, Vol. 16, No. 4, pp. 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, 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. 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. [Electronic resource]. URL: https://www.ibm.com/products/hpc-workload-management (accessed October 15, 2023).
8. TORQUE Resource Manager. [Electronic resource]. URL: http://www.adaptivecomputing.com/products/torque/ (accessed October 15, 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. [Online]. URL: https://www.zabbix.com/documentation/ (accessed October 15, 2023).
14. Grafana Labs. Grafana Documentation. [Online]. URL: https://grafana.com/docs/ (accessed October 15, 2023).
15. Elasticsearch B.V. Kibana Guide. [Electronic resource]. URL: https://www.elastic.co/guide/en/kibana/current/index.html (accessed October 15, 2023).
16. Masliy, A.A. “Creating a Web Interface for Monitoring the Status of Computing Clusters.” In: Engineering Thought. Proceedings of the II Republican Scientific and Practical Conference. Kazan, 2024. pp. 62–64.

© **А. А. Маслий** – студент каф. АССОИ, Казанский национальный исследовательский технологический университет (КНИТУ), Казань, Россия, anna.masliy.07@gmail.com; **Н. А. Староверова** – к.т.н., доц. каф. АССОИ КНИТУ, StaroverovaNA@corp.knrtu.ru.

© **A. A. Masliy –** Student, Department of Automated Systems for Data Collection and Processing (ASDCP), Kazan National Research Technological University (KNRTU), Kazan, Russia, anna.masliy.07@gmail.com; **N. A. Staroverova** – PhD (Chemical Sci.), Associate Professor, the ASDPO department, KNRTU, StaroverovaNA@corp.knrtu.ru.

Дата поступления рукописи в редакцию – 19.05.26.
Дата принятия рукописи в печать – 15.06.26.