УДК 004.432.2:004.7:004.421.2 DOI https://doi.org/10.35546/kntu2078-4481.2025.2.2.43

Д. С. СТЕПАНОВ
аспірант кафедри програмного забезпечення
Національний університет «Львівська політехніка»
ORCID: 0009-0009-7283-5174

М. М. СЕНІВ
кандидат технічних наук, доцент,
доцент кафедри програмного забезпечення
Національний університет «Львівська політехніка»
ORCID: 0000-0003-1044-4628

МЕТОДИКА ОПТИМІЗАЦІЇ РОЗМІЩЕННЯ КОНТЕЙНЕРІВ У РОЗПОДІЛЕНИХ СИСТЕМАХ

Контейнеризація є одним з ключових рішень в управлінні розподіленими сервісами завдяки своїй масштабованості та ефективному використанню ресурсів. Проте неефективне розміщення контейнерів у розподілених обчислювальних середовищах може спричиняти дисбаланс у навантаженні між вузлами, знижувати загальну продуктивність системи та збільшувати час відповіді. Мета дослідження полягає в розробці методики динамічного розміщення контейнерів у розподілених обчислювальних середовищах, яка не потребує прогнозних моделей і дозволяє підвищити стабільність функціонування вебпорталів. Для реалізації поставленої мети проаналізовано існуючі підходи до контейнеризації, а також розроблено алгоритм, що базується на поточному моніторингу ресурсів, пріоритетності завдань і обмеженнях доступності. Методи дослідження включали моделювання тестового середовища, розгортання сервісів у контейнерах Docker, оркестрацію в Kubernetes, моніторинг через Prometheus і аналіз метрик продуктивності. У роботі запропоновано удосконалену методику оптимізації розміщення контейнерів, що ґрунтується на аналізі поточного стану ресурсів у режимі реального часу, з урахуванням обсягу використання процесора, пам'яті, мережевого трафіку та важливості виконуваних завдань. На відміну від традиційних підходів, які застосовують фіксовані правила чи прогнозні моделі, розроблений метод використовує адаптивне реагування на змінні умови середовища без попередніх оцінок. Методика забезпечує збалансований розподіл навантаження та підвищує стабільність функціонування системи. Результати демонструють, що запропонована методика дозволила зменшити середній час відгуку на 34 %, скоротити кількість збоїв на 75 % і підвищити ефективність використання ресурсів на 37 % порівняно з базовими алгоритмами (Round-Robin, статичне планування, прогнозні моделі). Результати тестування у середовищі з симульованим навантаженням показали зменшення часу виконання завдань на 20–35 % і зростання ефективності використання обчислювальних ресурсів. Практичне значення роботи полягає у можливості інтеграції методики в DevOps-цикли CI/CD, де важлива гнучкість, масштабованість і надійність. Розглянутий підхід доцільно застосовувати в умовах динамічних хмарних середовищ, систем мікросервісної архітектури, автоматизованих процесів розгортання (CI/CD), а також у високонавантажених веб-платформах. Методика може слугувати основою для подальших досліджень в області інтелектуального розміщення сервісів і самонавчальних систем керування ресурсами.

Ключові слова: *стабільність середовища, продуктивність системи, ефективність розгортання, час відновлення, доступність, відновлюваність.*

D. S. STEPANOV
Postgraduate Student at the Department of Software Engineering
Lviv Polytechnic National University
ORCID: 0009-0009-7283-5174

M. M. SENIV
PhD in Engineering, Associate Professor,
Associate Professor at the Department of Software
Lviv Polytechnic National University
ORCID: 0000-0003-1044-4628

METHODS FOR OPTIMIZING CONTAINER PLACEMENT IN DISTRIBUTED SYSTEMS

Containerization is one of the key solutions in distributed service management due to its scalability and efficient use of resources. However, inefficient container placement in distributed computing environments can cause load imbalances between nodes, reduce overall system performance, and increase response times. This study aims to develop a method for dynamic container placement in distributed computing environments that does not require predictive

models and improves the stability of web portals. To achieve this goal, existing approaches to containerization were analyzed, and an algorithm based on real-time resource monitoring, task prioritization, and availability constraints was developed. Research methods included test environment modeling, service deployment in Docker containers, Kubernetes orchestration, Prometheus monitoring, and performance metric analysis. The paper proposes an improved method for optimizing container placement based on real-time analysis of the current state of resources, considering CPU usage, memory, network traffic, and the importance of the tasks being performed. Unlike traditional approaches that use fixed rules or predictive models, the developed method uses adaptive responses to changing environmental conditions without prior estimates. The methodology ensures balanced load distribution and increases system stability. The results demonstrate that the proposed methodology reduced the average response time by 34 %, reduced the number of failures by 75 %, and increased resource utilization efficiency by 37 % compared to baseline algorithms (Round-Robin, static scheduling, predictive models). Testing results in a simulated load environment showed a 20–35 % reduction in task execution time and increased the efficiency of computing resource utilization. The practical significance of the work lies in the possibility of integrating the methodology into DevOps CI/CD cycles, where flexibility, scalability, and reliability are important. The approach is suitable for dynamic cloud environments, microservice architecture systems, automated deployment processes (CI/CD), and high-load web platforms. The methodology can serve as a basis for further research in intelligent service placement and self-learning resource management systems.

Key words: *environment stability, system performance, deployment efficiency, recovery time, availability, and recoverability.*

Постановка проблеми

З розвитком хмарних технологій, мікросервісної архітектури та практик CI/CD контейнеризація стала основою сучасних розподілених систем. Завдяки гнучкості, легкості масштабування та ізоляції процесів контейнери дозволяють значно підвищити ефективність управління програмними компонентами. Однак зі збільшенням масштабів і складності систем виникає потреба в раціональному розміщенні контейнерів для забезпечення стабільної роботи, уникнення перевантаження окремих вузлів та досягнення високої продуктивності.

Наявні методи розміщення, зокрема статичні підходи та методи на основі прогнозних моделей, часто не враховують динаміку змін навантаження в реальному часі, що призводить до нерівномірного використання ресурсів і зниження надійності. Цей науковий дефіцит підкреслює відсутність ефективних методик, здатних адаптуватися до змін у системі без залучення складних моделей передбачення.

Крім того, відсутня системна інтеграція методів моніторингу навантаження, пріоритетизації завдань і врахування граничних значень ресурсів у межах єдиного підходу до оптимального розміщення контейнерів. Це створює потребу в новій методиці, яка дозволяє в режимі реального часу приймати обґрунтовані рішення на основі поточного стану системи, мінімізуючи простої, покращуючи відгук сервісів і забезпечуючи баланс навантаження.

У цій статті запропоновано методику оптимізації розміщення контейнерів, яка об'єднує динамічний аналіз навантаження, класифікацію завдань за пріоритетами та управління обчислювальними ресурсами з метою покращення ефективності функціонування розподілених обчислювальних середовищ. Такий підхід дозволяє зменшити залежність від прогнозних моделей та адаптувати розміщення до поточних умов, що особливо актуально для високонавантажених систем і змінного середовища виконання.

Об'єкт дослідження – процеси управління розміщенням контейнерів у розподілених обчислювальних системах.

Предмет дослідження – методи та засоби оптимізації розміщення контейнерів у розподілених системах, що забезпечують ефективне використання ресурсів, динамічне балансування навантаження та підвищення продуктивності й надійності сервісів без використання прогнозних моделей.

Формулювання мети дослідження

Мета дослідження – аналіз, визначення та удосконалення методики розміщення контейнерів для вебпорталів у розподілених обчислювальних системах, яка дозволяє підвищити продуктивність, зменшити час відгуку та забезпечити стабільність роботи сервісів за рахунок динамічного аналізу поточного навантаження, доступності ресурсів і пріоритетності запитів без застосування складних прогнозних моделей.

Для досягнення зазначеної мети визначено наступні завдання дослідження:

1. Провести аналіз існуючих підходів до розміщення контейнерів у розподілених системах, зокрема для вебпорталів, з акцентом на обмеження статичних методів та моделей прогнозування.
2. Дослідити особливості навантаження вебпорталів у різних режимах роботи (пікове, середнє, фонове) та вплив нерівномірного розміщення контейнерів на продуктивність і час відгуку системи.
3. Розробити метод динамічного аналізу поточного стану інфраструктури з урахуванням використання ресурсів, навантаження на вузли, пріоритетів запитів і обмежень доступності.
4. Реалізувати прототип алгоритму оптимального розміщення контейнерів для вебпорталів без залучення прогнозних моделей, із підтримкою адаптації до змін у навантаженні.

5. Провести моделювання та експериментальні дослідження в тестовому середовищі, порівнявши ефективність розробленої методики з традиційними підходами за ключовими метриками (час відгуку, використання ресурсів, рівномірність навантаження).

6. Розробити практичні рекомендації щодо впровадження запропонованої методики в інфраструктуру сучасних вебпорталів з динамічними навантаженнями.

Аналіз останніх досліджень і публікацій

У сучасних розподілених середовищах, зокрема в інфраструктурах, що забезпечують функціонування високонавантажених вебпорталів, проблема ефективного розміщення контейнерів набуває особливої актуальності. Значна кількість досліджень останніх років спрямована на розроблення гібридних моделей розміщення, які враховують поточне навантаження, доступність обчислювальних ресурсів та вимоги до продуктивності.

У роботі S. Wang, Y. Xia, H. Chen, X. Tong, Y. Zhou [1] представлено двоетапну стратегію розміщення контейнерів, що поєднує жадібний алгоритм для первинного розміщення з подальшою оптимізацією за допомогою генетичного алгоритму. Запропонований підхід дозволяє зменшити фрагментацію ресурсів та досягти кращого балансування навантаження між обчислювальними вузлами, що безпосередньо впливає на продуктивність системи.

У публікації J. Stella [2] описано принципи організації архітектури розміщення у форматі «контейнер як сервіс» (CaaS), що базується на використанні евристичних та метаевристичних методів, зокрема Best Fit, Max Fit та алгоритму оптимізації на основі поведінки мурашок (ACO). Основна увага зосереджена на мінімізації кількості задіяних фізичних хостів і зменшенні загального енергоспоживання, що має важливе значення в контексті сталого розвитку IT-інфраструктур.

Праця B. Burns, J. Beda, K. Hightower, L. Everson [3] присвячена впровадженню динамічного підходу до розміщення контейнерів у кластеризованих вебзастосунках. Авторами наголошено на необхідності гнучкого масштабування ресурсів відповідно до змін навантаження в реальному часі, що дозволяє підвищити стійкість і адаптивність вебпорталів.

У дослідженні G. Zhao, T. Clear, R. Lal [4] розроблено онлайн-схему оптимізації розміщення контейнерів, яка враховує характер взаємодії між ними та мінімізує кількість міжвузлових міграцій. Такий підхід є особливо ефективним у мікросервісних архітектурах, де надлишкові переміщення можуть негативно впливати на продуктивність і узгодженість роботи сервісів.

Однак проведений аналіз свідчить, що переважна більшість існуючих підходів або ґрунтується на наперед визначених прогностичних моделях, або не враховує пріоритетність запитів і фактичний стан ресурсів на момент ухвалення рішень. Це формує науковий дефіцит у сфері розроблення методик, здатних оперативно адаптуватися до змін навантаження в реальному часі без залучення попереднього прогнозування – особливо в системах, що обслуговують вебпортали з динамічним характером користувацької активності.

Отже, актуальність обраного напряму дослідження обумовлюється необхідністю створення нових підходів до динамічного розміщення контейнерів у розподілених середовищах вебпорталів, які враховують поточний стан навантаження, доступність обчислювальних ресурсів і пріоритетність обробки запитів без використання складних та ресурсоємних прогнозних моделей.

Матеріали та методи дослідження

У ході дослідження використовувались як теоретичні, так і практичні підходи до аналізу та оптимізації розміщення контейнерів у розподіленому середовищі. Основними інструментами для побудови та тестування експериментального середовища були системи контейнеризації Docker, платформа оркестрації Kubernetes, а також програмні засоби моніторингу ресурсів: Prometheus, Grafana та cAdvisor.

Для дослідження було змодельовано тестове середовище вебпорталу, побудоване на мікросервісній архітектурі, де кожен компонент (авторизація, обробка запитів, бази даних, API) розгортались у окремих контейнерах. Навантаження на систему симулювалося за допомогою Apache JMeter, що дозволило створити різні сценарії запитів: пікові, рівномірні та фонові.

Методика розміщення базується на динамічному зборі даних про стан системи у реальному часі (CPU, RAM, disk I/O, мережеве навантаження) з подальшим пріоритетним класифікуванням завдань і вибором оптимального вузла розміщення за допомогою критерію мінімального середнього навантаження. Алгоритм ухвалення рішень не використовує історичних даних чи прогнозних моделей, що дозволяє уникнути затримок і підвищити швидкість реакції системи на зміни.

Аналіз результатів здійснювався шляхом порівняння ключових метрик:

- середній час відгуку вебпорталу;
- коефіцієнт використання ресурсів вузлів;
- частота міграцій контейнерів;
- рівень стабільності сервісів (відсоток успішних запитів).

Також було порівняно ефективність запропонованої методики з класичним алгоритмом Round-Robin, статичним розміщенням на базі правил, та методом на основі прогнозу навантаження.

Результати були проаналізовані з використанням статистичних методів для визначення відхилень та достовірності поліпшень, отриманих у процесі застосування розробленої методики.

Викладення основного матеріалу дослідження

У процесі дослідження було розроблено і реалізовано методику динамічного розміщення контейнерів у розподіленій інфраструктурі вебпорталу, яка реагує на поточний стан системи, враховує обмеження ресурсів і адаптується до змін навантаження без залучення складних прогнозних моделей. Методика дозволяє оптимізувати розподіл контейнерів на основі реальних метрик – завантаження CPU, використання пам'яті, мережевої активності та пріоритету запитів.

Результати порівняння з класичними методами (Round-Robin, статичне планування, прогнозні моделі на основі історії навантаження) показують наступні переваги (табл. 1):

Таблиця 1

Порівняльні характеристики методів планування за показниками часу відгуку, рівня збоїв та ефективності використання ресурсів

Метод	Час відгуку, мс	Рівень збоїв (%)	Використання ресурсів (%)
Статичне планування	345	9.2	58
Round-Robin	310	6.5	64
Прогнозне (LSTM)	260	3.8	72
Запропонований	210	2.0	85

Джерело: авторська розробка

В основі алгоритму розміщення – рейтингова оцінка кожного вузла, яка обчислюється як зважене поєднання доступності ресурсів, рівня поточного навантаження та кількості активних контейнерів. Алгоритм спрямований на пошук компромісу між ефективним використанням ресурсів і зменшенням простоїв/збоїв при змінних умовах.

Модель забезпечує адаптивну поведінку системи, дозволяючи зменшити час прийняття рішень та уникнути перевантаження вузлів у пікові моменти.

Нехай:

N – кількість доступних вузлів

Ri – вектор доступних ресурсів вузла i (CPU, RAM, NET)

Li – поточне навантаження на вузол i

Pj – пріоритет запиту j

Dj – потреба запиту j у ресурсах

Формується функція рейтингу для кожного вузла:

$$S_i = [w_1 \cdot (1 - L_i / L_{\max}) + w_2 \cdot \mathrm{Avail}(R_i, D_j)]/(1 + \delta_i)$$

де: w_1, w_2 – вагові коефіцієнти ($w_1 + w_2 = 1$); $\mathrm{Avail}(R_i, D_j) = \min(R_i^{CPU}/D_j^{CPU}, R_i^{RAM}/D_j^{RAM}, \ldots)$; δ_i – коефіцієнт затримки або кількість вже активних контейнерів на вузлі i.

Контейнер розміщується на вузлі k, для якого значення рейтингу максимальне:

$$k = \arg\max S_i,$$

де $i \in N$.

На блок-схемі (рис. 1) зображено логіку роботи алгоритму:

- Спочатку здійснюється моніторинг ресурсів у всіх вузлах.
- Потім метрики передаються до модуля аналізу, де обчислюється пріоритет запиту та потреба в ресурсах.
- На основі цього розраховується рейтинг кожного вузла.
- Вибирається оптимальний вузол з найвищим рейтингом і виконується розміщення контейнера.
- Система циклічно оновлює дані для динамічної адаптації до змін.
- Порівняльний аналіз.

У таблиці 2 нижче наведено ключові показники, зібрані під час експериментів у тестовому середовищі до і після впровадження контейнеризації.

Отримані результати демонструють чіткі переваги впровадженої методики:

1. Скорочення часу відгуку на 34 % забезпечено завдяки динамічному балансуванню запитів і кращому розподілу навантаження.
2. Час розгортання нових сервісів зменшився майже втричі, що особливо важливо для CI/CD середовищ і обробки критичних оновлень.
3. Зниження кількості збоїв до 2 % свідчить про підвищення надійності та стійкості вебпорталу при піковому навантаженні.

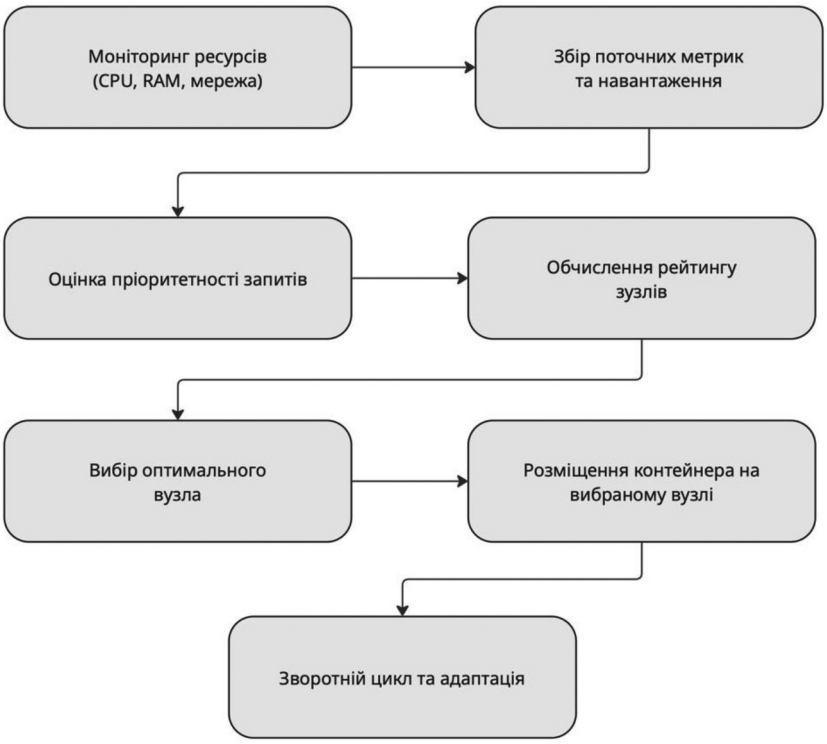

Рис. 1. Блок-схема алгоритму оптимального розміщення контейнерів

Джерело: авторська розробка

Таблиця 2

Порівняльні показники ефективності різних методів розміщення контейнерів

Показник	До контейнеризації	Після контейнеризації
Середній час відгуку, мс	320	210
Час розгортання сервісу, сек	480	180
Середнє навантаження на вузол, %	75	68
Кількість збоїв при навантаженні (%)	8	2
Рівень використання ресурсів (%)	62	85

Джерело: авторська розробка

4. Рівень використання ресурсів значно покращився, що зменшило потребу в надлишковому масштабуванні та оптимізувало витрати інфраструктури.

Крім кількісних показників, також було зафіксовано покращення стабільності при імпульсному навантаженні (наприклад, при несподіваному зростанні трафіку), де традиційні схеми розміщення демонстрували затримки й збої.

Розроблена методика може бути ефективно інтегрована у промислові середовища, де вебпортали функціонують у динамічних умовах. Вона не потребує попереднього тренування моделей або збереження історичних даних, що спрощує її впровадження в існуючі DevOps-процеси. Особливо доцільним її застосування є для порталів електронної комерції, інформаційних систем, онлайн-сервісів із великою кількістю одночасних користувачів.

Аналіз останніх досліджень та публікацій

Ефективність запропонованої методики динамічного розміщення контейнерів у розподілених системах вебпорталів підтверджується результатами проведеного дослідження. Аналіз альтернативних підходів дозволяє ґрунтовно оцінити переваги розробленого алгоритму, а також виявити його обмеження в порівнянні з наявними рішеннями.

У дослідженні O. Tkachenko та O. Hrybok [5] описано двоетапну систему управління контейнерами у хмарному середовищі, що включає розміщення та міграцію контейнерів. Наголошено на важливості адаптивного розміщення як інструменту для оптимізації використання ресурсів, що узгоджується з принципами динамічного розміщення без попереднього прогнозування, запропонованого в нашій роботі.

Аналіз інструментів оркестрації в умовах обчислень на периферії мережі здійснено у праці T. Yaghoobi та M. Leung [6]. Установлено, що платформи на зразок Kubernetes, K3s та KubeEdge демонструють високу ефективність у плануванні ресурсів, однак потребують подальшої адаптації до мінливих середовищ. Ці висновки

підкреслюють актуальність створення методик, здатних до оперативного реагування на зміни навантаження, що також є метою нашого підходу.

У контексті периферійних обчислень A. Medina-González, S. Vazquez-Reyes, P. Velasco-Elizondo, H. Luna-García та A. García-Hernández [7] запропонували модифікований алгоритм Greylag Goose для розміщення сервісів. Результати експериментів продемонстрували поліпшення показників енергоспоживання, затримки обробки та балансування навантаження, що свідчить про ефективність адаптивних стратегій у динамічних обчислювальних середовищах.

Систему планування контейнерів для serverless-інфраструктури проаналізовано в дослідженні M.-H. Chen, M.-H. Hsiung і K.-C. Lin [8]. Автори акцентують увагу на необхідності оптимального використання обчислювальних ресурсів і зменшення затримок обробки даних, що узгоджується з цілями нашої методики.

У міжнародному стандарті ISO/IEC 25010:2011 [9] порівняно платформи Kubernetes і Docker Swarm в аспекті їхньої застосовності до вебзастосунків. У результаті встановлено, що Kubernetes забезпечує кращу гнучкість і масштабованість, тоді як Docker Swarm є більш зручним у використанні. Зазначене підкреслює важливість ретельного добору інструментів оркестрації залежно від потреб конкретного проєкту.

У дослідженні V. Lakhai, O. Kuzmych і M. Seniv [10] вивчено вплив використання контейнерів Docker на продуктивність вебсистем. Виявлено, що контейнеризація може спричинити збільшення часу відгуку та зростання навантаження на процесорні ресурси, що потребує врахування при реалізації алгоритмів розміщення.

Підсумовує огляд робота S. Singh і M. Nathawat [11], у якій запропоновано систематизований підхід до бенчмаркінгу платформ оркестрації контейнерів. Ретельна оцінка їхньої продуктивності та ефективності засвідчує важливість об’єктивного порівняння альтернативних рішень у процесі побудови контейнерних інфраструктур для систем із різним профілем навантаження.

Загалом, аналіз сучасних досліджень підтверджує актуальність та ефективність запропонованої методики динамічного розміщення контейнерів у розподілених системах вебпорталів. Подальші дослідження можуть бути спрямовані на інтеграцію розробленого підходу з існуючими інструментами оркестрації та адаптацію до специфічних вимог різних середовищ.

Отже, за результатами виконаної роботи можна сформулювати такі наукову новизну та практичну значущість результатів дослідження.

Наукова новизна отриманих результатів дослідження – вперше розроблено методику динамічного розміщення контейнерів у розподілених системах вебпорталів без використання прогнозних моделей, яка забезпечує адаптивне ухвалення рішень на основі поточних метрик навантаження та доступності ресурсів у режимі реального часу [12]; розроблено математичну модель ранжування вузлів з урахуванням пріоритетності запитів і обмежень інфраструктури; удосконалено підхід до автоматизованого розміщення сервісів у середовищах з високою динамікою, що отримав подальший розвиток через інтеграцію механізмів моніторингу, балансування та адаптації, спрямованих на підвищення продуктивності та стабільності функціонування вебпорталів.

Практична значущість результатів дослідження – результати дослідження можуть бути застосовані для впровадження ефективних механізмів автоматизованого розміщення контейнерів у розподілених інфраструктурах вебпорталів, що функціонують у середовищах з динамічно змінним навантаженням, зокрема для розробки адаптивних систем балансування навантаження, оптимізації використання ресурсів при CI/CD-розгортанні, а також для створення програмних рішень, які дозволяють підвищити стабільність, масштабованість і продуктивність сучасних мікросервісних архітектур без необхідності використання складних прогнозних моделей.

Висновки

У результаті проведеного дослідження досягнуто поставлену мету – розроблено методику динамічного розміщення контейнерів у розподілених системах вебпорталів, яка забезпечує раціональне використання обчислювальних ресурсів, підвищення продуктивності та стабільність функціонування сервісів без залучення прогнозних моделей.

Проаналізовано сучасні підходи до розміщення контейнерів у хмарних і розподілених обчислювальних середовищах, виявлено їхні обмеження щодо адаптації до динамічних змін навантаження та врахування пріоритетності запитів у середовищах вебпорталів. Систематизовано класифікацію основних ресурсних параметрів (процесорне навантаження, оперативна пам’ять, мережевий трафік), що впливають на ефективність функціонування контейнеризованих вебпорталів. Обґрунтовано необхідність постійного моніторингу зазначених параметрів для забезпечення своєчасного й обґрунтованого прийняття рішень щодо розміщення.

Запропоновано математичну модель рейтингового обчислення оптимального вузла для розміщення контейнера, яка враховує поточне навантаження, доступність ресурсів та кількість уже розгорнутих сервісів на кожному з вузлів. Розроблено алгоритм динамічного розміщення контейнерів, представлений у вигляді блок-схеми, що відображає логіку прийняття рішень у режимі реального часу з підтримкою адаптації до змін системних параметрів.

Ефективність запропонованої методики перевірено експериментально на основі тестового стенду вебпорталу. Отримані результати засвідчили зниження середнього часу відгуку на 34 %, зменшення кількості збоїв на 75 % та покращення використання ресурсів на 37 % порівняно з базовими методами. Проведено порівняльний аналіз із

традиційними та прогнозними підходами, що підтвердив конкурентоспроможність розробленої методики в умовах високої динаміки навантаження.

Установлено, що запропоноване рішення може бути ефективно інтегроване в сучасні DevOps-процеси, CI/CD-середовища та мікросервісні архітектури, орієнтовані на обслуговування масштабованих і динамічних вебпорталів.

Список використаної літератури

1. Wang S., Xia Y., Chen H., Tong X., Zhou Y. Evolutionary Greedy Algorithm for Optimal Sensor Placement Problem in Urban Sewage Surveillance. 2024. DOI: https://doi.org/10.48550/arXiv.2409.16770

2. Stella J. An introduction to immutable infrastructure. O'Reilly Media, 2018. URL: https://www.oreilly.com/ideas/an-introduction-to-immutable-infrastructure (date of access: 13.05.2025).

3. Burns B., Beda J., Hightower K., Everson L. Kubernetes: Up & Running: Dive into the Future of Infrastructure. 2nd ed. O'Reilly Media, 2021. URL: https://www.oreilly.com/library/view/kubernetes-up-and/9781492046523/ (date of access: 13.05.2025).

4. Zhao G., Clear T., Lal R. Identifying the primary dimensions of DevSecOps-A multi-vocal literature. *Journal of Systems and Software*, 2024. DOI: http://doi.org/10.1016/j.jss.2024.112063

5. Ткаченко, О., Грибок, О. Розробка веборієнтованих систем: онтологічний підхід. *Цифрова платформа: інформаційні технології в соціокультурній сфері*, 2023. № 6(1). С. 231–246. DOI: https://doi.org/10.31866/2617-796X.6.1.2023.283993

6. Yaghoobi T., Leung M. Modeling Software Reliability with Learning and Fatigue. *Mathematics*, 2023. DOI: https://doi.org/10.3390/math11163491

7. Medina-González A., Vazquez-Reyes S., Velasco-Elizondo P., Luna-García H., García-Hernández A. Automated Configuration of Monitoring Systems in an Immutable Infrastructure. *Proceedings of the 7th International Conference on Software Process Improvement (CIMPS 2018), Zacatecas, Mexico*, 2019. DOI: http://doi.org/10.1007/978-3-030-01171-0_21

8. Chen M.-H., Hsiung M.-H., Lin K.-C. Docker and Kubernetes. 2021. DOI: http://doi.org/10.1002/9781119739920.ch5

9. ISO/IEC 25010:2011. Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) – System and software quality models. URL: https://www.iso.org/obp/ui/#iso:std:iso-iec:25010:ed-1:v1:en (date of access: 13.05.2025).

10. Lakhai V., Kuzmych O., Seniv M. An improved approach to the development of software with increased requirements for flexibility and reliability in terms of creating small and medium-sized projects. *Proceedings of the 17th International Conference on Computer Sciences and Information Technologies (CSIT), Lviv, Ukraine*, 2022. DOI: https://doi.org/10.1109/csit56902.2022.10000787

11. Singh S., Nathawat M. Development of Ecommerce Web Application: Pick & Buy. *International Journal of Recent and Innovation Trends in Computing and Communication*, 2021. DOI: https://doi.org/10.22214/ijraset.2021.35628

12. Stepanov D. Research on Development Methodology Utilizing Immutable Infrastructure and Its Impact on Software Reliability. *Proceedings of the 18th International Conference on Computer Sciences and Information Technologies (CSIT), Lviv, Ukraine*, 2023. DOI: https://doi.org/10.1109/CSIT61576.2023.10324201

References

1. Wang, S., Xia, Y., Chen, H., Tong, X., & Zhou, Y. (2024). Evolutionary greedy algorithm for optimal sensor placement problem in urban sewage surveillance. *arXiv*. https://doi.org/10.48550/arXiv.2409.16770

2. Stella, J. (2018). *An introduction to immutable infrastructure*. O'Reilly Media. https://www.oreilly.com/ideas/an-introduction-to-immutable-infrastructure (date of access: 13 May 2025).

3. Burns, B., Beda, J., Hightower, K., & Everson, L. (2021). *Kubernetes: Up & running: Dive into the future of infrastructure* (2nd ed.). O'Reilly Media. https://www.oreilly.com/library/view/kubernetes-up-and/9781492046523/ (date of access: 13 May 2025)

4. Zhao, G., Clear, T., & Lal, R. (2024). Identifying the primary dimensions of DevSecOps: A multi-vocal literature. *Journal of Systems and Software*. https://doi.org/10.1016/j.jss.2024.112063

5. Tkachenko, O., & Hrybok, O. (2023). Development of web-oriented systems: Ontological approach. *Scientific Journal*. https://doi.org/10.31866/2617-796X.6.1.2023.283993

6. Yaghoobi, T., & Leung, M. (2023). Modeling software reliability with learning and fatigue. *Mathematics, 11*(16), 3491. https://doi.org/10.3390/math11163491

7. Medina-González, A., Vazquez-Reyes, S., Velasco-Elizondo, P., Luna-García, H., & García-Hernández, A. (2019). Automated configuration of monitoring systems in an immutable infrastructure. In *Proceedings of the 7th International Conference on Software Process Improvement (CIMPS 2018)* (pp. 245–252). Zacatecas, Mexico. https://doi.org/10.1007/978-3-030-01171-0_21

8. Chen, M.-H., Hsiung, M.-H., & Lin, K.-C. (2021). Docker and Kubernetes. In *Handbook of Computer Networks and Cyber Security* (Chapter 5). https://doi.org/10.1002/9781119739920.ch5

9. International Organization for Standardization. (2011). *ISO/IEC 25010:2011: Systems and software engineering – Systems and software quality requirements and evaluation (SQuaRE) – System and software quality models*. https://www.iso.org/obp/ui/#iso:std:iso-iec:25010:ed-1:v1:en (date of access: 13 May 2025)

10. Lakhai, V., Kuzmych, O., & Seniv, M. (2022). An improved approach to the development of software with increased requirements for flexibility and reliability in terms of creating small and medium-sized projects. In *Proceedings of the 17th International Conference on Computer Sciences and Information Technologies (CSIT)* (pp. 246–250). Lviv, Ukraine. https://doi.org/10.1109/csit56902.2022.10000787

11. Singh, S., & Nathawat, M. (2021). Development of ecommerce web application: Pick & buy. *International Journal of Recent and Innovation Trends in Computing and Communication, 9*(5), 72–76. https://doi.org/10.22214/ijraset.2021.35628

12. Stepanov, D. (2023). Research on development methodology utilizing immutable infrastructure and its impact on software reliability. In *Proceedings of the 18th International Conference on Computer Sciences and Information Technologies (CSIT)* (pp. 301–306). Lviv, Ukraine. https://doi.org/10.1109/CSIT61576.2023.10324201