– $G(c_{T3}) = s_{T3}$: лог-сервер збирає 150 000 подій на добу, журнали зберігаються 3 років.

Крок 4. Перевірка узгодженості (композиція $H = G \circ F$).

Перевіряємо, чи відповідає фактичний стан вимогам:

– $H(r_{T1}) = s_{T1}$: шифрування реалізовано відповідно до вимог;

– $H(r_{T2}) = s_{T2}$: MFA та RBAC налаштовано коректно;

– $H(r_{T3}) = s_{T3}$: логи зберігаються 3 років (менше ніж мінімум 6), що не відповідє вимозі.

Виявлена невідповідність: логи зберігаються менше 6 років.

Дія: система сповіщає системного адміністратора про невідповідність.

Отже, запропонована категорно-функторна модель дала змогу формалізувати вимоги та обмеження до проєктування MIC у приватних хмарних середовищах у вигляді математично узгоджених категорій об’єктів і морфізмів, що усуває неоднозначність на етапах формування вимог.

Таким чином, розроблена категорно-функторна модель формалізації вимог до хмарних інфраструктур MIC забезпечує математично обґрунтовану основу для автоматизованого управління конфігураціями IT-інфраструктури. Її застосування дозволяє здійснювати перевірку узгодженості, своєчасно виявляти невідповідності між вимогами та фактичним станом системи і мінімізувати ризики виникнення розбіжностей у конфігурації системи.

Список використаної літератури

[1] A. Pesqueira, M. J. Sousa, and R. Pereira, “Individual dynamic capabilities and artificial intelligence in health operations: Exploration of innovation diffusion,” Intelligence-Based Medicine, vol. 11, p. 100239, 2025. DOI: 10.1016/j.ibmed.2025.100239

[2] R. Wan, “Research on Application Configuration Management Technology of Distributed Cloud Platform,” IEEE Access, 2022. DOI: 10.1109/ACCESS.9836624. [Online]. Available: https://ieeexplore.ieee.org/document/9836624/

[3] C. Dickerson and M. Wilkinson, “Architecture, analysis and design of systems using extensions of category theory,” IEEE Open Journal of Systems Engineering, pp. 1–14, 2024, DOI: 10.1109/OJSE.2024.3432570.

УДК 004.05

АЛГОРИТМІЧНІ ПРИНЦИПИ ОПТИМІЗАЦІЇ ОБРОБКИ КОРИСТУВАЦЬКИХ ЗАПИТІВ В REST API ВЕБЗАСТОСУНКАХ

Чередниченко М. Є. Літвінова Ю. С. (maksym.cherednychenko@cs.khpi.edu.ua, uliya.litvinova@khpi.edu.ua)

Національний технічний університет "Харківський політехнічний інститут" (Україна)

У тезах розглядається проблема обробки великої кількості запитів авторизованих користувачів у вебзастосунках, що призводить до перевантаження серверів і зниження продуктивності. Пропонується покращення принципів проєктування REST API шляхом їх розширення для підвищення ефективності обробки користувацьких запитів.

В умовах зростання кількості користувачів вебзастосунків важливою проблемою стає збільшення кількості одночасних запитів. Це створює проблему перевантаження серверів, збільшення затримок та блокування ресурсів на тривалий час. Зокрема, коли запити користувачів потребують авторизації й охоплюють значні обсяги даних, система змушена витрачати ресурси на постійну перевірку доступу до даних користувачів, що негативно впливає на загальну продуктивність і стабільність роботи.

Для ефективного вирішення цієї проблеми пропонується розширити принципи REST API [1], впровадивши два додаткових принципи, які допомагають оптимізувати обробку запитів і знизити навантаження на серверну частину.

REST API описує ресурси та їх обробку і передбачає принцип безстанового оброблення запитів (stateless), коли сервер не зберігає стан користувача між запитами, і кожен запит містить всю необхідну інформацію для своєї обробки [2]. Для цього REST допускає використання токенів,

ключів API або інших методів контролю доступу, ці підходи залишаються поза стандартом і реалізуються переважно на рівні конкретної системи. У результаті сервер часто витрачає ресурси на перевірку прав доступу навіть для операцій, які могли б бути безпечними і публічними. З огляду на це було розроблено два додаткових принципи.

Перший запропонований додатковий принцип визначає чітке розділення публічних та приватних ресурсів. Якщо ресурс є публічним, доступ до нього повинен бути відкритим без токенів або авторизаційних механізмів. Авторизація застосовується лише для приватних або персоналізованих даних. Серверна частина не повинна перевіряти авторизацію без потреби.

Застосування цього принципу дозволяє значно знизити навантаження на серверну частину та підвищити швидкодію системи, адже кількість складних перевірок доступу суттєво зменшується. Також розділення запитів підвищує прозорість архітектури, оскільки розробники та адміністратори чітко визначають, які частини даних є загальнодоступними, а які потребують захисту.

Другий принцип визначає порядок обробки запитів користувачів і може базуватися на класифікації запитів з попереднього. При проєктуванні API запити повинні оброблятися з урахуванням їх критичності для користувача та системи. Критичні запити (наприклад, доступ до актуальних публічних або приватних даних) отримують вищий пріоритет, тоді як менш важливі або ресурсоємні запити обробляються відкладено або з обмеженнями. Впровадження такого принципу дозволяє забезпечити більш ефективне управління ресурсами сервера і покращити досвід користувачів за рахунок зменшення затримок для критичних операцій.

Зростаюча актуальність проблеми перенавантаження серверів вимагає обов’язкових дій щодо її вирішення. Розширення принципів REST API за рахунок чіткого розділення публічних і приватних ресурсів та пріоритизації запитів дозволяє зменшити навантаження на сервери і скоротити час обробки великої кількості критичних запитів. Таким чином, доповнені принципи REST формалізують обов’язкові правила роботи з даними та створюють основу для більш ефективного та керованого оброблення запитів користувачів у сучасних вебзастосунках.

Список використаної літератури

[1] “What is REST API?” visual-paradigm.com. Available: https://www.visual-paradigm.com/guide/development/what-is-rest-api/ [Accessed: October 03, 2025].
[2] L. Richardson and S. Ruby, RESTful Web Services. Sebastopol, CA: O’Reilly Media, 2007, pp. 79-106.

УДК 519.87

КОМП’ЮТЕРНЕ МОДЕЛЮВАННЯ “ВИСОКООРГАНІЗОВАНОГО” БОЮ З ГРУПОВИМ УРАЖЕННЯМ ЦІЛІ В СТОХАСТИЧНІЙ ПОСТАНОВЦІ У PYTHON

Черновол Н.М., Бобрицька Г.С., Фурсенко О.К.
(n.n.chernovol@gmail.com, bogalina31@ukr.net, Olfursenko@ukr.net)
Харківський національний університет Повітряних Сил ім. І. Кожедуба (Україна)

Побудована комп’ютерна модель “високоорганізованого” бою з урахуванням умови: одним влучним пострілом по цілі вражається декілька бойових одиниць противників бойового протистояння. Ця модель представлена алгоритмом розв’язання задачі в стохастичній постановці за допомогою мови програмування Python.

Дослідження моделей бойових дій, обґрунтованих математично, почалось з роботи Ф. Ланчестера [1]. По сьогоднішній день ці моделі застосовуються авторами інших наукових робіт. Причому ускладнення постановок задач або доведення задачі до кінцевих розрахунків привело до застосування комп’ютерних пакетів або програм, наприклад в публікаціях [2,3]. Автори в своїх роботах для реалізації методу динамічного програмування в задачах моделювання бойових дій оптимізаційного характеру застосовують пакет Mathematika [4], мову програмування Python [5]; для реалізації розв’язання задачі в стохастичній постановці використовують Matlab [6].