УДК 004.42

ВИЯВЛЕННЯ ТА АНАЛІЗ АНТИПАТЕРНІВ У СЕРВІС-БАЗОВАНИХ ІНФОРМАЦІЙНИХ СИСТЕМАХ

Сусляков А.В. – гр. МгКІ-1-24, магістрант, andreilunev2708@gmail.com
Пісоцький А.В. – PhD, ст.викл., pisotskyi.av@knutd.edu.ua
Київський національний університет технологій та дизайну

Метою роботи є виявлення та систематизація антипатернів, характерних для сервіс-базованих інформаційних систем (SOA) для підвищення якості, надійності та ефективності програмних рішень.

Сучасні інформаційні системи все частіше реалізуються у вигляді сервіс-базованих архітектур, що забезпечують масштабованість, гнучкість і повторне використання компонентів. Водночас зростає складність керування взаємодією сервісів, їхньою еволюцією та моніторингом, що зумовлює появу антипатернів - типових помилок проєктування чи інтеграції, які призводять до зниження продуктивності та надійності системи [1].

Серед найпоширеніших антипатернів у SOA можна виділити такі:

- Service Chain Overload – надмірна послідовність викликів сервісів, що збільшує час відповіді.
- Chatty Service – надто багато дрібних запитів між сервісами.
- Shared Database Integration – спільна база даних кількох сервісів, що ускладнює масштабування.
- God Service – надмірна концентрація функцій у одному сервісі.
- Versioning Chaos – хаотичне управління версіями сервісів без узгодженої стратегії.

На *рис. 1* наведено узагальнену схему процесу виявлення антипатернів у сервіс-базованій архітектурі.

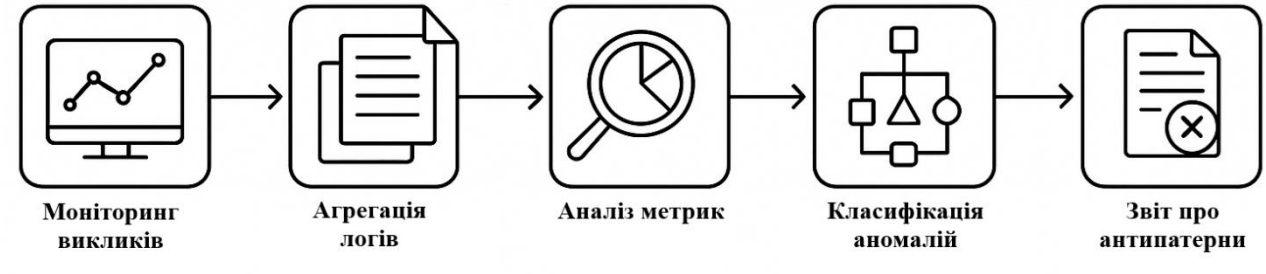

Рисунок 1 – схема процесу виявлення антипатернів у сервіс-базованій системі

Виявлення антипатернів здійснюється на архітектурному, кодовому та поведінковому рівнях. На архітектурному рівні аналізують зв’язність компонентів, кількість залежностей і складність викликів. На рівні коду

проводять статичний аналіз для виявлення дублювання логіки, надмірного використання ресурсів і порушень принципів SOLID. Поведінковий рівень охоплює моніторинг сервісів, аналіз логів і часу відповіді, що дає змогу виявити приховані архітектурні проблеми [2].

Для кількісного аналізу використовують метрики архітектурної складності: Cyclomatic Complexity, Service Coupling Index, Message Exchange Frequency, Average Response Time [3]. Вони дають змогу оцінити взаємозалежність сервісів і визначити потенційні зони ризику. У *таблиці 1* подано приклади поширених антипатернів та їхніх наслідків.

Таблиця 1 – Характеристика антипатернів у сервіс-базованих системах

Назва антипатерну	Причина виникнення	Наслідок для системи	Можливий метод усунення
Service Chain Overload	Надмірна декомпозиція процесів	Зростання часу відгуку	Оптимізація ланцюгів викликів
Chatty Service	Часті дрібні запити між сервісами	Перевантаження мережі, затримки	Використання пакетної передачі даних
Shared Database Integration	Порушення принципу ізоляції даних	Проблеми оновлення та тестування	Розподіл бази на незалежні схеми
God Service	Централізація бізнес-логіки	Зниження гнучкості, складність підтримки	Рефакторинг та розподіл функцій
Versioning Chaos	Відсутність контролю версій	Несумісність сервісів, помилки обміну	Впровадження політики версіонування

Висновок. В результаті проведеного аналізу встановлено що антипатерни є невід’ємною складовою життєвого циклу сервіс-базованих інформаційних систем. Їх своєчасне виявлення дозволяє мінімізувати технічний борг, забезпечити стабільність роботи сервісів і підвищити рівень автоматизації управління архітектурою.

Список використаних джерел:

1. Ford N., Richards M., Sadalage P., Dehghani Z. Software Architecture: The Hard Parts. Boston : O’Reilly Media, 2021. 459 p.

2. Hohpe G., Woolf B. Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions. Boston : Addison-Wesley, 2003. 650 p.

3. Гавриленко С. І. Архітектурні патерни та антипатерни програмних систем: монографія. Київ : Наук. думка, 2020. 120 с.