Жиляєв Є. В., магістрант,
Вакалюк Т. А., д-р. пед. наук, проф., професор кафедри ІПЗ
Державний університет «Житомирська політехніка»

ВІДСТЕЖЕННЯ ПОДІЙ У РОЗПОДІЛЕНИХ КОМП'ЮТЕРНИХ СИСТЕМАХ

З точки зору як історії так і дизайну, початковим етапом розробки програмного забезпечення була розробка програм як єдиного цілого – монолітної архітектури, коли усе від графічних елементів до зв'язку із базою даних є елементами одного програмного юніту. Такий підхід має певну кількість недоліків, тим не менш, у деяких випадках таке архітек-турне рішення є необхідністю. Розробники програмного забезпечення без чіткого розуміння та загально визначеного рішення намагалися подолати ці недоліки своїми методами доки не було винайдено загальноприйнятне рішення – мікросервісна архітектура, дизайн який скла-дається з деякої кількості незалежних сервісів. Деякі з проблем, які виникають внаслідок монолітичної архітектури є: збільшений об'єм коду, коли з додаванням нових можливостей програми також збільшується складність та заплутаність коду, що також ускладнює внесення будь-яких подальших змін.

Написання великих програм із використанням мікросервісної архітектури зустріло позитивні відгуки, однак, немає чітких контур того, що саме складає мікросервісний стиль і як його притримуватись. Більш того, така архітектура має свої унікальні проблеми. Запит кожного користувача, який надходить до одного мікросервісу, майже завжди перенаправляється від одного сервісу до іншого, кожен з яких може бути написаний за допомогою різних фреймворків та програмних мов. Це ставить нову задачу перед командою розробки – як зрозуміти і відстежити подорож кожного такого запиту. Ця задача складається із багатьох факторів, таких як вищезгадана неоднорідна природа мікросервісів, сама природа розподільних систем, в яких запиту доводиться подорожувати через мережу від сервісу до сервісу без будь-яких гарантій щодо стану мережі майбутніх сервісів. Усе це і вимагає дослідження та оптимізації доступних методів та інструментів, які можуть допомогти у вирішені вищезгаданої задачі.

Перехід на мікросервісну архітектурну успішно відбувається як для великих, так і для малих компаній. Згідно з опитуванням, проведеним O'Reilly у 2020 році, наразі наявні наступні тренди:

– Практики пов'язані з мікросервісною архітектурою становляться все більш зрілими. Близько 28% респондентів відповіли, що їх організація використовує мікросервіси щонайменше три роки; більш ніж три п'ятих (61%) респондентів використовують мікросервіси не менш ніж рік.

– Фірми мають великі плани на мікросервіси. Майже одна третя (29%) респондентів відповіли, що їх наймачі мігрують або ж імплементують більшість їх систем (понад 50%) на основі мікросервісної архітектури.

– Більшість фірм, які перейшли на мікросервіси позитивно оцінюють свій досвід. Менш ніж 10% доповіли про "повний успіх", але більшість (54%) описують свій досвід принаймні як "достатньо успішний" та 92% респондентів мали хоч якійсь успіх.

– Успіх з мікросервісами можна напряму пов'язати з "володінням" усього життєвого циклу програмного забезпечення. Більшість респондентів (74%) відповіли, що саме їх команди володіють циклом "збірка-тестування-розгортання-підтримка". Такі команди є на 18% більш успішними, ніж ті, що відмовилися від такої практики.

– Зростаюча складність є найбільшим викликом при переході на мікросервісну архітектуру. Якщо поєднати в опитувані "Збільшена складність системи" та "Складність управління багатьма сервісами", можна побачити, що саме "складність = складання за багатьох елементів" є найбільшим викликом, який згадується 56% респондентів.

Як можна побачити з опитування від O'Reilly, мікросервісна архітектура є дуже успішною і популярною практикою, яка використовується все більшою кількістю команд, при цьому головним викликом для таких команд є саме зростаюча складність мікросервісних архітектур, а найбільш успішними командами є ті, які володіють усім життєвим циклом програмного забезпечення, зокрема займаються підтримкою своїх рішень. Отже, відслідковування процесів та їх моніторинг у таких розподілених системах є критичним фактором задля їх успішного використання. Проблема складності та розуміння системи, над якою працюєш, занадто часто недооцінюється і дуже багато команд в певний момент знаходять себе у ситуації коли теперішня система стала занадто складною, відслідковування процесів є неможливим, внаслідок чого, рішення кожної помилки може призвести до неочікуваних результатів. Задля продовження цього дослідження з більш практичної перспективи, буде розглянута банківська система оповіщень, побудована на основі розподіленої мікросервісної архітектури. В ній будуть використані такі інструменти, як Zipkin та Brave OpenTracing задля відстеження процесів, збору та аналізу усіх супутніх даних.