ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

УДК 004.94:004.65:629.3.027 **doi: 10.32620/aktt.2025.5.09**

А. І. МАЛЮГА

Національний аерокосмічний університет «Харківський авіаційний інститут», Харків, Україна

ГІБРИДНА ПРОГРАМНА АРХІТЕКТУРА З ПЕРИФЕРІЙНИМИ ОБЧИСЛЕННЯМИ ДЛЯ АДАПТИВНИХ VR-СИСТЕМ НАВЧАННЯ ВОДІННЮ З БІОМЕТРИЧНИМ ЗВОРОТНИМ ЗВ'ЯЗКОМ

Предметом статті є методологія розробки гібридної трирівневої програмної архітектури для адаптивних VR-систем навчання водінню, з біометричним зворотним зв'язком у реальному часі, що забезпечує розподіл функціональних компонентів між клієнтським, периферійним та мікросервісним рівнями. Метою є розробка архітектури, яка забезпечує баланс між мінімальною затримкою обробки біометричних даних для комфортного VR-досвіду, зниженням витрат, та можливістю поступового масштабування. Завдання, що потребують вирішення: виконання порівняльного аналізу існуючих архітектурних підходів за критеріями латентності, вартості та масштабованості; розробка гібридної моделі програмної архітектури; експериментальна перевірка запропонованої архітектури; визначення області застосування гібридної архітектури. Застосовані методи включають методи архітектурного проектування, методи синтезу та декомпозиції. Були отримані наступні результати. Розроблено гібридну архітектуру з клієнтським рівнем для рендерингу та збору біометричних даних, периферійним рівнем для класифікації стану користувача та адаптації навчальних сценаріїв, й мікросервісним рівнем для асинхронної симуляції трафіку і аналітики даних з можливим масштабуванням. Експериментальна перевірка продемонструвала зниження латентності обробки біометричних даних. Визначено сферу застосування розробленої гібридної моделі. Висновки. Результати дослідження підтвердили актуальність гібридного підходу до архітектурного проектування з розміщенням обчислювально інтенсивних компонентів на периферійному сервері в локальній мережі навчального закладу, замість централізованого хмарного сервера, для усунення затримок глобальної мережі, а також дали можливість обґрунтувати вибір програмної архітектури з урахування кількості користувачів, бюджетних обмежень та вимог до латентності. Наукова новизна отриманих результатів полягає у створенні моделі гібридної архітектури, що передбачає розподіл функціональних компонентів системи по рівням, на основі обмежень щодо швидкості обробки, вартості впровадження та ступеня незалежності допоміжних обчислень, що дає можливість підвищити ефективність використання ресурсів при навчанні із застосуванням технології віртуальної реальності, а також підвищити ефективність навчання, за рахунок адаптації до поточного стану особи, яка навчається.

Ключові слова: *адаптивна віртуальна реальність; біометричний зворотний зв'язок; навчання водіїв; периферійні обчислення; гібридна архітектура; оптимізація затримки; машинне навчання; мікросервіси; масштабованість; класифікація напружень.*

1. Вступ

Системи віртуальної реальності (VR-системи) набули широкого застосування у підготовці водіїв, завдяки можливості створення безпечного контрольованого середовища для відпрацювання навичок керування, без виникнення ризиків реальних дорожніх ситуацій [1]. Сучасні VR-системи навчання водінню еволюціонують від статичних тренажерів у напрямку створення адаптивних систем, що інтегрують дані з біометричних сенсорів безперервного моніторингу фізіологічного стану учня та на їх основі динамічно коригують рівень складності навчальних сценаріїв в реальному часі, відповідно до поточного стану водія [2]. Такий індивідуалізований підхід значно підвищує ефективність процесу засвоєння навичок безпечного водіння, порівняно зі статичними тренажерами [2].

Проте, проектування програмної архітектури адаптивних VR-систем з біометричним зворотним зв'язком стикається з наявністю суперечливих вимог [3]. Обробка біометричних даних та адаптація VR-сцени повинні відбуватися з мінімальною затримкою, оскільки надмірна латентність порушує відчуття присутності у віртуальному середовищі [4, 5]. Водночас обмежені бюджети навчальних закладів вимагають розроблення економічно ефективних рішень, що можуть масштабуватися для використання як у невеликих автошколах, так і в більших за масштабом навчальних центрах [4, 6].

Існуючі архітектурні підходи мають суттєві обмеження з урахуванням наведених у [7, 8] вимог. Монолітна архітектура забезпечує найнижчу латентність, завдяки відсутності мережевих затримок, але вимагає при цьому застосування високовартісного обладнання на кожному робочому місці та ускладнює централізоване управління [8]. Клієнт-серверна архітектура дає можливість знизити вартість клієнтського обладнання, але при її застосуванні суттєво зростає латентність через затримки глобальної мережі, при цьому часто перевищується комфортний поріг для VR-досвіду [9] Периферійні обчислення пропонують компромісне рішення через локальну обробку в межах навчального закладу, проте мають обмежену потужність для ресурсоємних задач симуляції автомобільного трафіку [9]. Мікросервісна архітектура забезпечує масштабованість, але додає накладні витрати на оркестрацію множини сервісів, що критично для систем реального часу [4].

Тобто, існуючі підходи не задовольняють комплексним вимогам щодо мінімальної латентності для отримання комфортного VR-досвіду, економічної доступності для навчальних закладів з бюджетними обмеженнями, можливості поступового масштабування без значних матеріальних перевитрат, а також можливості централізованої аналітики прогресу учнів у навчання. Таке протиріччя обґрунтовує актуальність розробки гібридної архітектури, що поєднує переваги кількох підходів на основі розподілу функціональних компонент системи віртуальної реальності для навчання водіїв між декількома рівнями програмної системи з урахуванням протиріч у обмеженнях. Відсутність таких архітектурних рішень обмежує використання адаптивних VR-технологій для формування навичок безпечного водіння на основі індивідуалізованого навчання з урахуванням психофізіологічного стану учня.

1.1. Мотивація дослідження

Розвиток адаптивних VR-систем для навчання водінню з біометричним зворотним зв'язком стримується відсутністю архітектурних рішень, що забезпечують баланс між продуктивністю та економічною доступністю для навчальних закладів, що готують майбутніх водіїв [3, 9]. Існуючі архітектурні підходи мають недоліки, що обмежують їх впровадження в умовах обмежених бюджетів українських автошкіл. [10-15].

Монолітна архітектура, де всі компоненти системи – VR-рендеринг, обробка біометричних даних, машинне навчання для класифікації стресу учня, симуляція реалістичного трафіку, централізована аналітика – виконуються на кожному клієнтському комп'ютері, забезпечує найнижчу латентність обробки біометричних даних у діапазоні 80-120 мс завдяки повній відсутності мережевих затримок [10]. Проте, така архітектура вимагає високовартісного обладнання на кожному робочому місці, що робить систему економічно недоступною для більшості навчальних закладів та унеможливлює поступове масштабування від кількох робочих місць до десятків без значних додаткових витрат. Також проблемою є ускладнене централізоване оновлення моделей машинного навчання та навчальних сценаріїв, оскільки кожен клієнт потребує окремого розгортання, що збільшує витрати часу на супровід системи [12, 13].

Клієнт-серверна архітектура радикально знижує вартість клієнтського обладнання через перенесення всіх обчислень на централізований сервер з потоковою передачею відео до тонких клієнтів. Однак латентність обробки біометричних даних зростає до 205-475 мс через накладні витрати глобальної мережі при комунікації з хмарними провайдерами: типова мережева затримка становить 50-150 мс в одну сторону, подвоюючись для двостороннього обміну, до чого додаються затримки кодування та декодування відеопотоку [14]. В [3] експериментально показано що централізований рендеринг у хмарі призводить до латентності, яка перевищує комфортний поріг для VR-додатків, негативно впливаючи на якість навчального досвіду. Крім того, централізований сервер створює єдину точку відмови – його збій блокує навчальний процес для всіх учнів одночасно, знижуючи надійність системи [8].

Периферійні обчислення пропонують компромісне рішення, розміщуючи обробку біометричних даних та ML-вивід на локальному сервері в межах навчального закладу з підключенням через високошвидкісну локальну мережу. Це знижує мережеву латентність до 15-25 мс порівняно з 50-150 мс для глобальної мережі, забезпечуючи сумарну латентність обробки біометрії 95-200 мс [14]. У роботі [11] автори експериментально підтвердили, що периферійна обробка для VR-застосунків знижує 95-й перцентиль латентності на 42-55% порівняно з хмарною обробкою при збереженні високої якості досвіду користувача. Проте периферійний сервер обмежений обчислювальною потужністю для ресурсоємних задач, зокрема симуляції реалістичного трафіку з великою кількістю автономних агентів, що критично для створення складних навчальних сценаріїв [15, 16].

Мікросервісна архітектура розділяє систему на незалежні сервіси з можливістю індивідуального масштабування через автоматичне балансування навантаження в хмарному середовищі. В [12] показали, що мікросервісна архітектура забезпечує лі-

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

нійну масштабованість для великої кількості одночасних користувачів при збереженні високої пропускної здатності. Однак оркестрація множини мікросервісів додає 25-75 мс латентності через накладні витрати на міжсервісну комунікацію, синхронізацію стану та розподілене трасування запитів, що є критичним для систем реального часу з обробкою біометричних даних [17].

Аналіз існуючих підходів виявляє, що жоден з них окремо не задовольняє комплексним вимогам адаптивних VR-систем для навчальних закладів. Монолітна архітектура мінімізує латентність за рахунок економічно неприйнятної високої вартості кожного робочого місця. Клієнт-серверна знижує вартість обладнання за рахунок надмірної латентності, що перевищує комфортний поріг для VR-досвіду. Периферійні обчислення балансують латентність та вартість, але обмежені обчислювальною потужністю для важких задач симуляції. Мікросервісна забезпечує гнучку масштабованість, але додає критичні накладні витрати на оркестрацію для систем реального часу.

Розробка гібридної архітектури, що комбінує переваги кількох підходів на основі обґрунтованого розподілу функціональних компонентів між клієнтським, периферійним та мікросервісним рівнями, дозволить досягти скорочення циклу розробки та впровадження систем віртуальної реальності для підготовки водіїв, створення низьковартісної початкової конфігурації системи з можливістю поступового масштабування, а також забезпечити підтримку адаптивного навчання з урахуванням психофізичного стану особи, яка навчається.

1.2. Огляд літератури

Підходи до побудови програмної архітектури VR-систем для навчання водінню включають монолітні архітектури з локальною обробкою на клієнтських комп'ютерах, клієнт-серверні архітектури з централізованим рендерингом, периферійні обчислення з локальними серверами в межах навчального закладу, та мікросервісні архітектури з розподіленою обробкою через множину незалежних сервісів. Кожний підхід має специфічні переваги та обмеження щодо латентності обробки, вартості обладнання та масштабованості системи.

У роботі [3] авторами запропоновано гібридний підхід CloudVR з передбаченням у хмарі та корекцією на периферії, де хмарний сервер виконує основний рендеринг, а периферійний пристрій коригує позиціювання об'єктів на основі локальних сенсорних даних. Експериментальна перевірка продемонструвала істотне покращення затримки motion-to-photon на 58% при збереженні високої якості рендерингу. Проте підхід орієнтований на індивідуальне використання та не розглядає масштабованість для багатокористувацького навчального середовища.

У роботі [9] авторами проведене експериментальне порівняння периферійних та хмарних VR-застосунків у аспекті отриманого досвіду користувача. Результати тестування підтвердили, що периферійна обробка забезпечує значно вищу якість досвіду порівняно з хмарною обробкою, знижуючи латентність на 42-55%. Критичною метрикою виявилася стабільність частоти кадрів, нестабільність якої істотно погіршує сприйняття користувача. Дослідження підтверджує переваги периферійних обчислень, проте не формалізують критерії розподілу функціональних компонентів між архітектурними рівнями.

В публікації [12] наведено результати дослідження ефекту досягнення непомітної затримки в мобільному хмарному геймінгу, що має схожу проблематику з VR-системами. Результати показали, що розділений рендеринг між клієнтом та хмарою суттєво знижує сприйману затримку порівняно з повним хмарним рендерингом. Однак, для VR-систем такий підхід має обмеження через необхідність стереоскопічного рендерингу з урахуванням відстеження положення голови.

Адаптивні VR-системи з біометричним зворотним зв'язком характеризуються інтеграцією фізіологічних сенсорів для безперервного моніторингу стану учня, алгоритмами класифікації стресу та когнітивного навантаження на основі біометричних даних, та механізмами динамічної адаптації складності навчальних сценаріїв у реальному часі відповідно до виявленого стану користувача [18, 19].

Авторами статті [1] описана система оцінки поведінки водіїв у VR-симуляторі з використанням носимих сенсорів для моніторингу частоти серцевих скорочень та гальванічної реакції шкіри. Дослідження виявило значущу кореляцію між варіабельністю серцевого ритму та помилками водіння ($r = 0.67$, $p < 0.001$), що підтверджує можливість використання біометричних даних для оцінки стану водія. Система використовує класифікацію на основі машинного навчання для категоризації рівня стресу, проте не адаптує навчальний сценарій у реальному часі.

У роботі [2] запропонована архітектура адаптивної VR-системи з фізіологічною адаптацією на основі даних електроенцефалографії, відстеження погляду та серцевого ритму. Алгоритм адаптації використовує навчання з підкріпленням для динамічного налаштування параметрів складності, включаючи щільність трафіку, погодні умови та часові обмеження. Експериментальна валідація продемонстру-

вала покращення результатів навчання на 31% порівняно з неадаптивною системою. Однак система потребує інвазивних сенсорів, що обмежує практичне застосування в автошколах.

В роботі [7] описана адаптивна VR-система з інтерфейсом мозок-комп'ютер та відстеженням погляду для тренування складних візуально-моторних задач. Система адаптує складність інтерфейсу на основі індикаторів когнітивного навантаження, отриманих з біометричних даних. Результати підтверджують суттєве скорочення часу навчання порівняно з неадаптивними системами. Архітектура використовує локальну обробку всіх сигналів на високопродуктивному обладнанні, не розглядаючи економічно ефективні альтернативи для навчальних закладів з обмеженими бюджетами.

Мікросервісні архітектури для розподілених VR-застосунків забезпечують гнучку масштабованість через декомпозицію системи на незалежні сервіси, кожний з яких може масштабуватися індивідуально відповідно до навантаження, та використання стандартизованих протоколів міжсервісної комунікації для забезпечення інтероперабельності.

В [4] досліджували динамічне розміщення мікросервісів для доставки VR-контенту в середовищі периферійних обчислень. Запропонований алгоритм оптимізації використовує цілочисельне лінійне програмування для мінімізації латентності з урахуванням обмежень на обчислювальні ресурси. Симуляційні експерименти продемонстрували істотне зниження середньої затримки порівняно з простими евристичними алгоритмами при збереженні обмежень на якість обслуговування. Проте дослідження обмежується комп'ютерним моделюванням без реальної перевірки на фізичній інфраструктурі.

Авторами публікації [12] проаналізовані продуктивність та масштабованість мікросервісної архітектури для високопродуктивних застосунків. Порівняльне тестування різних протоколів міжсервісної комунікації виявило, що gRPC забезпечує 2.3× нижчу латентність порівняно з REST API при високому навантаженні. Дослідження підкреслює важливість вибору протоколу комунікації для систем реального часу.

Розподілені симуляції трафіку характеризуються декомпозицією обчислювального навантаження симуляції між множиною обчислювальних вузлів, механізмами синхронізації стану симуляції між розподіленими компонентами, та підтримкою великої кількості автономних агентів для створення реалістичних дорожніх сценаріїв.

В статті [5] описаний розподілений механізм безпеки для симуляції автономних транспортних засобів. Архітектура розділяє симуляцію на множину екземплярів з синхронізацією через протокол розподіленої інтерактивної симуляції. Експерименти показали, що розподілена симуляція забезпечує суттєве прискорення порівняно з монолітною симуляцією при збереженні детермінізму. Проте підхід потребує складної синхронізації годинників та вирішення конфліктів для областей перекриття.

В [13] представили SUMO - відкритий мікроскопічний симулятор міського руху з підтримкою великої кількості транспортних засобів. Симулятор забезпечує детальну симуляцію на рівні смуг руху з моделями слідування за автомобілем, зміни смуги та логіки світлофорів. Інтеграція з іншими симуляторами через інтерфейс ко-симуляції дозволяє гібридні сценарії з комбінацією детальної та спрощеної симуляції для оптимізації продуктивності.

Реалізація таких застосунків забезпечується з використання контейнеризації [20, 21], а оптимізація роботи – з урахуванням мережного трафіку [22, 23].

Периферійні обчислення для низько-латентних застосунків характеризуються розміщенням обчислювальних ресурсів у безпосередній близькості до користувачів, зниженням мережевої затримки через обробку в локальній мережі замість глобальної, та розподіленою архітектурою з множиною периферійних вузлів для забезпечення масштабованості [5].

Дослідження періоду 2019-2024 років підтверджують, що периферійні обчислення на рівні базових станцій забезпечують низьку затримку обробки для застосунків Інтернету речей та промислової автоматизації, що свідчить про доцільність розгортання периферійних серверів при достатній кількості одночасних користувачів [3, 19].

Сучасні патерни масштабованості для розподілених систем включають архітектуру на основі подій з незмінними журналами, розділення відповідальності команд та запитів для оптимізації обробки, та патерн розподілених транзакцій для забезпечення узгодженості даних. Ці патерни забезпечують горизонтальну масштабованість та відмовостійкість через використання незмінних журналів подій та евентуальної узгодженості. Проте патерни додають архітектурну складність та потребують експертизи в DevOps [11].

Існуючі підходи до проектування програмних архітектур адаптивних VR-систем для навчання водінню з біометричним зворотним зв'язком мають ряд недоліків, зокрема відсутність формалізованих моделей розподілу функціональних компонентів між архітектурними рівнями з критеріями оптимізації, брак експериментальних досліджень оптимального співвідношення клієнтів до периферійного сервера на основі багатокритеріального аналізу, відсутність економічних моделей з аналізом точки беззбитковості для порівняння різних архітектурних підходів, та недостатність практичних рекомендацій

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

щодо вибору архітектури з урахуванням компромісів між латентністю, вартістю та складністю інтеграції для навчальних закладів різного масштабу. Таким чином, важливою є задача розробки науково обґрунтованої гібридної архітектури з формалізованими критеріями розподілу компонентів, експериментальною перевіркою та економічними обмеженнями для забезпечення балансу між продуктивністю та доступністю системи.

1.3. Мета та завдання дослідження

Метою роботи є розробка гібридної трирівневої програмної архітектури адаптивної VR-системи навчання водінню з біометричним зворотним зв'язком у реальному часі, що забезпечує відповідає вимогам щодо затримок обробки, обмеженої вартості та масштабованості на основі розподілу функціональних компонентів між клієнтським, периферійним та мікросервісним рівнями.

1) Провести багатокритеріальний порівняльний аналіз архітектурних підходів для адаптивних VR-систем навчання водінню з біометричним зворотним зв'язком за критеріями латентності обробки даних, вартості розгортання та масштабованості. Аналіз має виявити компроміси кожного підходу та обґрунтувати необхідність гібридної архітектури.

2) Розробити модель програмної архітектури, що забезпечує розподіл функціональних модулів адаптивної VR-системи між трьома архітектурними рівнями з критеріями задоволення обмежень щодо затримки рендерингу для клієнтського рівня, зменшення мережевої латентності та забезпечення конфіденційності біометричних даних для периферійного рівня.

3) Виконати експериментальну перевірку запропонованої архітектури шляхом перевірки латентності обробки біометричних даних для різних сценаріїв навантаження, економічної ефективності з порівнянням до альтернативних підходів, та масштабованості.

4) Визначити область застосування гібридної архітектури залежно від потреб навчального закладу та з урахуванням обмежень щодо латентності, вартості та складністю інтеграції.

Стаття структурована наступним чином. У розділі 2 представлено гібридну трирівневу архітектуру адаптивної VR-системи з описом розподілу функціональних компонентів по рівням архітектури. У розділі 3 описано методологію експериментальної перевірки запропонованої архітектури. У розділі 4 наведено результати експериментальних досліджень латентності, економічної ефективності та масштабованості. У розділі 5 наведено обговорення отриманих результатів у порівнянні з альтернативними архітектурними підходами. У розділі 6 підсумовуються результати роботи, сформульовано висновки та представлено напрямки майбутніх досліджень.

2. Гібридна програмна архітектура з периферійними обчисленнями для адаптивних VR-систем

Запропонована гібридна програмна архітектура адаптивної VR-системи навчання водінню з біометричним зворотним зв'язком має три функціональні рівні: клієнтський рівень, периферійний рівень та мікросервісний рівень. Формально архітектура представлена як кортеж із елементів трьох рівнів:

$$A=<C,E,M>$$

де C – клієнтський рівень з компонентами візуалізації та введення даних, E – периферійний рівень з компонентами обробки біометрії та адаптації, M – мікросервісний рівень з компонентами асинхронної обробки та аналізу.

Клієнтський рівень виконує функції, що безпосередньо впливають на якість VR-досвіду: стереоскопічний рендеринг віртуальної сцени з частотою 90-120 кадрів за секунду, відстеження положення голови у шести ступенях свободи, збір біометричних даних зі носимих пристроїв через бездротовий зв'язок, та обробку користувацького вводу для керування віртуальним автомобілем. Розміщення цих компонентів на клієнті мінімізує затримку між рухом користувача та відображенням відповідної зміни у віртуальному середовищі, що критично для запобігання кіберзапаморочення.

Периферійний рівень розміщується на локальному сервері в межах навчального закладу з підключенням через високошвидкісну локальну мережу. Цей рівень виконує обчислювально-інтенсивні функції, що потребують низької затримки: класифікацію рівня стресу на основі біометричних даних через алгоритми машинного навчання, розрахунок параметрів адаптації навчального сценарію відповідно до виявленого стану користувача, та управління станом віртуальної сцени через інтерфейси взаємодії з рушіями візуалізації. Периферійне розміщення забезпечує конфіденційність біометричних даних через обробку в локальній мережі без передачі в зовнішню хмару, а також спільне використання обчислювальних ресурсів для групи клієнтів одночасно, що знижує вартість на одного учня через ефект масштабу.

Мікросервісний рівень представлений як набір контейнеризованих сервісів у хмарному середовищі з автоматичним керуванням масштабуванням. Цей

рівень виконує асинхронні обчислювально важкі задачі, що не є критичними для затримки віртуального рендерингу: симуляцію реалістичного трафіку з великою кількістю автономних агентів, збір та аналітику даних навчальних сесій для оцінки прогресу учнів, генерацію персоналізованих навчальних сценаріїв на основі історичних даних. Асинхронна комунікація через черги повідомлень дозволяє відокремити затримки цих задач від критичного шляху обробки біометрії, забезпечуючи еластичне масштабування на основі поточного рівня навантаження.

Особливості рівнів запропонованої системи наведено в табл. 1.

Таблиця 1

Особливості трирівневої програмної архітектури VR-системи

Рівень архітектури	**Функціональні компоненти**	**Основні функції компонентів**	**Масштабування**
Клієнтський рівень	Рушій віртуального рендерингу	Візуалізація тривимірної сцени, відстеження положення голови	Лінійне відносно кількості учнів
	Модуль збору біометричних даних	Постійний збір інформації щодо частоти серцевих скорочень та гальванічної реакції шкіри зі смарт-годинника через бездротовий зв’язок	
	Обробник користувацького вводу	Трансляція команд керування віртуальним автомобілем від периферійних пристроїв вводу	
	Мережевий клієнт	Серіалізація біометричних даних та синхронна передача на периферійний сервер	
Периферійний рівень	Класифікатор машинного навчання	Визначення рівня стресу користувача на основі статистичних характеристик біометричних даних	Розділене: один сервер на групу клієнтів
	Механізм адаптації	Розрахунок параметрів адаптації модифікації навчального сценарію згідно психофізіологічного стану учня	
	Інтерфейс управління віртуальною сценою	Синхронізація параметрів адаптації з рушієм візуалізації на клієнті	
	Аналізатор продуктивності	Моніторинг метрик системи та виявлення вузьких місць обробки	
Мікросервісний рівень	Сервіс симуляції трафіку	Обчислення руху автономних агентів у віртуальному середовищі з великою кількістю транспортних засобів	Еластичне, через автоматичне керування масштабуванням
	Сервіс аналітики даних	Агрегація метрик навчальних сесій та обчислення показників ефективності засвоєння навичок	
	Сервіс генерації сценаріїв	Створення персоналізованих навчальних завдань на основі історичних даних прогресу учня	
	Сховище даних	Збереження структурованих та неструктурованих даних з резервним копіюванням	

Розподіл функціональності базується на трьох критеріях, що задають обмеження для кожного рівня архітектури:

$L_{total}(C,E)$ – загальна латентність обробки біометричних даних, $C_{total}(N)$ – загальна вартість розгортання для N учнів, I(M) – ступінь ізоляції мікросервісного рівня від критичного шляху.

Перший критерій орієнтований на підтримку розміщення компонентів критичного шляху на клієнті та периферійному сервері таким чином, щоб зменшити сумарну затримку при обробці. Другий критерій дає можливість обґрунтувати централізацію обчислювально інтенсивних компонентів на сервері для спільного використання ресурсів між групою учнів. Третій критерій призначений для відбору некритичних для затримки задач та їх переносу на окремий мікросервісний рівень для незалежного масштабування без впливу на ефективність процесу адаптації системи в цілому.

Загальна латентність обробки біометричних

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

даних у запропонованій гібридній архітектурі складається з затримок клієнтського та периферійного рівнів, оскільки мікросервісний рівень функціонує асинхронно і не впливає на критичний шлях адаптації віртуальної сцени. Формула декомпозиції латентності має вигляд:

$$L_{total} = L_{client} + L_{network} + L_{edge'},$$

де L_{client} – латентність клієнтського рівня (збір біометрії та серіалізація даних), $L_{network}$ – мережева затримка передачі даних через локальну мережу, L_{edge} – латентність периферійного рівня (класифікація стресу, адаптація, оновлення віртуальної сцени).

Латентність клієнтського рівня визначається затримкою збору біометричних даних зі смарт-годинника через бездротовий зв'язок та серіалізації структурованих даних перед передачею на периферійний сервер. Латентність периферійного рівня включає десеріалізацію біометричних даних, виконання виводу класифікації через натреновану модель машинного навчання, обчислення параметрів адаптації навчального сценарію, та підготовку команд оновлення віртуальної сцени для передачі клієнту. Мережева латентність залежить від характеристик локальної мережі та є значно нижчою порівняно з глобальною мережею завдяки розміщенню периферійного сервера в межах навчального закладу.

Загальна вартість розгортання гібридної архітектури для групи учнів визначається сумою вартості клієнтського обладнання, периферійних серверів з урахуванням розділення навантаження, та операційних витрат мікросервісного рівня. Порівняльний аналіз вартості різних архітектурних підходів $C_{monolithic}(N)$ для монолітної архітектури для N учнів, $C_{client\text{-}server}(N)$ для клієнт-сервеної архітектури, $C_{hybrid}(N,K)$ – для гібризної архітектури формалізується через наступні співвідношення:

$$C_{monolithic}(N)=NC_{high\text{-}end},$$
$$C_{client\text{-}server}(N)= C_{server}+NC_{thin},$$
$$C_{hybrid}(N,K)= NC_{mid}+(N/K)C_{edge}+C_{cloud}(N),$$

де $C_{high\text{-}end}$ – вартість високопродуктивного обладнання для монолітної архітектури, C_{server} – вартість централізованого сервера, C_{thin} – вартість тонкого клієнта, C_{mid} – вартість клієнта середнього рівня продуктивності, C_{edge} – вартість периферійного сервера, K – кількість клієнтів на один периферійний сервер, $C_{cloud}(N)$ – операційні витрати хмарних мікросервісів.

Гібридна архітектура забезпечує нижчу вартість на одного учня порівняно з монолітною через спільне використання периферійного сервера групою клієнтів, що представляється як:

$$C_{hybrid}(N,K)/N < C_{monolithic}(N)/N = C_{high\text{-}end},$$

за умови $C_{mid} + C_{edge}/K + C_{cloud}(N)/N < C_{high\text{-}end}$, що виконується при достатньо великому значенні K через ефект масштабу.

Клієнтський рівень виконує функції, критичні для забезпечення комфортного досвіду віртуальної реальності, що вимагають мінімальної затримки між діями користувача та відповідною зміною віртуального середовища. Основними функціональними компонентами цього рівня є рушій віртуального рендерингу, модуль збору біометричних даних, обробник користувацького вводу та мережевий клієнт для комунікації з периферійним сервером.

Рушій віртуального рендерингу забезпечує стереоскопічну візуалізацію тривимірної сцени з частотою 90-120 кадрів за секунду для кожного ока окремо, що необхідно для створення відчуття присутності у віртуальному середовищі. Компонент реалізує відстеження положення голови користувача у шести ступенях свободи через вбудовані інерційні датчики та зовнішні системи позиціонування, забезпечуючи синхронне оновлення точки огляду у віртуальній сцені. Рендеринг віртуального автомобіля, дорожньої інфраструктури та трафіку виконується з урахуванням фізично коректного освітлення та тіньових ефектів для підвищення реалістичності навчального середовища. Функціональність може бути реалізована з використанням сучасних рушіїв ігрової індустрії, таких як Unity або Unreal Engine.

Модуль збору біометричних даних виконує безперервний моніторинг фізіологічного стану учня через інтеграцію зі смарт-годинником або іншими носимими пристроями. Компонент збирає частоту серцевих скорочень та гальванічну реакцію шкіри з частотою один вимір за секунду, формує ковзне вікно спостережень заданого розміру та обчислює статистичні характеристики біометричних сигналів для передачі на периферійний сервер. Вибір мобільних пристроїв обумовлений вимогами неінвазивності вимірювань та можливості використання в умовах навчальних закладів без потреби спеціалізованого медичного обладнання. Комунікація з носимими пристроями здійснюється через стандартні протоколи бездротового зв'язку низької енергії.

Обробник користувацького вводу трансформує команди від периферійних пристроїв керування у відповідні дії віртуального автомобіля. Компонент має підтримувати різноманітні пристрої вводу, включаючи кермо з педалями або спрощені інтерфейси з клавіатурою та мишею для економічних конфігурацій. Обробка вводу виконується з мінімальною затримкою для забезпечення природної реакції віртуального автомобіля на команди користувача.

Мережевий клієнт забезпечує комунікацію з периферійним сервером через синхронні виклики процедур для передачі біометричних даних та отримання параметрів адаптації навчального сценарію. Компонент виконує серіалізацію структурованих даних у компактний бінарний формат перед передачею для зменшення обсягу мережевого трафіку та затримки комунікації. Реалізація може використовувати сучасні протоколи віддаленого виклику процедур з ефективною серіалізацією, такі як gRPC з Protocol Buffers.

Латентність клієнтського рівня визначається затримкою збору біометричних даних зі смарт-годинника через бездротовий зв'язок та часом серіалізації структурованих даних перед передачею на периферійний сервер. Вартість клієнтського обладнання складається з вартості персонального комп'ютера середнього рівня продуктивності, гарнітури віртуальної реальності та смарт-годинника для збору біометричних даних.

Периферійний рівень реалізує централізацію обчислювально інтенсивних функцій обробки біометричних даних та адаптації навчальних сценаріїв на локальному сервері в межах навчального закладу. Розміщення сервера в локальній мережі забезпечує низьку мережеву затримку комунікації з клієнтами та конфіденційність обробки чутливих біометричних даних без передачі в зовнішню хмару.

Класифікатор машинного навчання виконує визначення рівня стресу користувача на основі статистичних характеристик біометричних сигналів. Компонент використовує попередньо натреновану модель машинного навчання, що приймає на вхід вектор ознак біометричних даних та повертає дискретний рівень стресу. Вибір алгоритму класифікації базується на компромісі між точністю передбачення та швидкістю виводу на серверному обладнанні. Модель тренується на зібраних даних з анотованими рівнями стресу, визначеними експертами або через об'єктивні метрики помилок водіння.

Механізм адаптації обчислює параметри модифікації навчального сценарію відповідно до виявленого психофізіологічного стану учня. Компонент реалізує систему на правилах або політику навчання з підкріпленням, що визначає зміни складності навчального завдання для оптимізації прогресу засвоєння навичок. При виявленні високого рівня стресу механізм знижує складність сценарію через зменшення щільності трафіку, зниження швидкості руху інших транспортних засобів або покращення погодних умов. При низькому рівні стресу складність підвищується для підтримання оптимального когнітивного навантаження.

Інтерфейс управління віртуальною сценою забезпечує синхронізацію параметрів адаптації з рушієм візуалізації на клієнті. Компонент генерує команди оновлення стану віртуального середовища та передає їх клієнту для застосування у наступному циклі рендерингу. Інтерфейс абстрагує специфіку різних рушіїв візуалізації, забезпечуючи можливість підтримки множини платформ без модифікації логіки адаптації.

Аналізатор продуктивності виконує моніторинг метрик системи для виявлення вузьких місць обробки та забезпечення дотримання вимог до затримки. Компонент збирає дані про утилізацію процесора та пам'яті, мережеву затримку комунікації з клієнтами, та пропускну здатність класифікатора машинного навчання. При перевищенні порогових значень генеруються сповіщення для адміністраторів системи.

Латентність периферійного рівня включає десеріалізацію біометричних даних, виконання виводу класифікації через натреновану модель, обчислення параметрів адаптації, та підготовку команд оновлення віртуальної сцени. Вартість периферійного сервера визначається потребою багатоядерного процесора для паралельної обробки запитів від множини клієнтів та достатнього обсягу оперативної пам'яті для завантаження моделей машинного навчання. Один периферійний сервер може обслуговувати групу клієнтів одночасно, що забезпечує ефект масштабу через спільне використання обчислювальних ресурсів.

Мікросервісний рівень реалізує функції, які не є критичними для затримки адаптації віртуальної сцени, але потребують значних обчислювальних ресурсів або еластичного масштабування залежно від навантаження. Компоненти цього рівня функціонують асинхронно відносно критичного шляху обробки біометрії, взаємодіючи з периферійним рівнем через черги повідомлень.

Сервіс симуляції трафіку виконує обчислення руху автономних агентів у віртуальному середовищі з великою кількістю транспортних засобів для створення реалістичних дорожніх ситуацій. Компонент реалізує мікроскопічну модель транспортного потоку з індивідуальним управлінням поведінкою кожного агента, включаючи моделі слідування за автомобілем, зміни смуги руху та реакції на світлофори. Симуляція виконується незалежно від частоти рендерингу віртуальної сцени на клієнті, з періодичною передачею оновленого стану трафіку на периферійний сервер для синхронізації з віртуальним середовищем. Функціональність може бути реалізована з використанням спеціалізованих симуляторів трафіку, таких як CARLA або SUMO.

Сервіс аналітики даних виконує агрегацію метрик навчальних сесій для оцінки прогресу засвоєння навичок безпечного водіння. Компонент обчислює показники ефективності навчання на основі кількос-

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

ті помилок водіння, середнього рівня стресу під час сесії, часу реакції на дорожні ситуації та покращення відносно попередніх сесій. Результати аналітики зберігаються для генерації звітів для інструкторів та персоналізації майбутніх навчальних сценаріїв.

Сервіс генерації сценаріїв створює персоналізовані навчальні завдання на основі історичних даних прогресу учня. Компонент використовує алгоритми навчання з підкріпленням для оптимізації складності сценаріїв з метою максимізації швидкості засвоєння навичок без перенавантаження користувача. Генерація виконується асинхронно між навчальними сесіями, з підготовкою оптимального сценарію до початку наступної сесії.

Сховище даних забезпечує збереження структурованих та неструктурованих даних системи з резервним копіюванням для відмовостійкості. Компонент зберігає профілі користувачів, метрики навчальних сесій, натреновані моделі машинного навчання та записи навчальних сесій для подальшого аналізу. Реалізація використовує реляційні бази даних для структурованих даних та об’єктне сховище для великих неструктурованих даних.

Комунікація між периферійним та мікросервісним рівнями реалізується через черги повідомлень для асинхронної обробки подій. Периферійний сервер публікує події початку та завершення навчальних сесій, зміни рівня стресу та інші значущі події у чергу повідомлень. Мікросервіси підписуються на релевантні типи подій та обробляють їх незалежно без блокування критичного шляху адаптації. Функціональність черги повідомлень може бути реалізована з використанням спеціалізованих систем обміну повідомленнями, таких як Apache Kafka або RabbitMQ.

Мікросервісний рівень не вносить додаткової латентності у критичний шлях обробки біометрії, оскільки функціонує асинхронно відносно адаптації віртуальної сцени. Вартість цього рівня визначається операційними витратами використання хмарної інфраструктури з автоматичним керуванням масштабуванням залежно від навантаження.

Взаємодія компонентів запропонованої гібридної архітектури реалізується через два основні потоки даних: синхронний потік обробки біометрії для адаптації віртуальної сцени у реальному часі та асинхронні потоки симуляції трафіку і аналітики даних, що не впливають на критичну затримку системи.

Синхронний потік обробки біометричних даних забезпечує адаптацію навчального сценарію відповідно до поточного психофізіологічного стану учня з мінімальною затримкою. Клієнтський рівень виконує безперервний збір фізіологічних сигналів зі смарт-годинника, формує статистичні характеристики сигналів за ковзним вікном спостережень та серіалізує дані у компактний формат для передачі. Серіалізовані біометричні характеристики передаються на периферійний сервер через синхронний виклик процедури класифікації стресу з обмеженням часу очікування відповіді. Периферійний сервер десеріалізує отримані дані, виконує вивід класифікації рівня стресу через натреновану модель машинного навчання та передає результат механізму адаптації. Механізм адаптації обчислює параметри модифікації навчального сценарію на основі виявленого стану користувача, формуючи команди зміни щільності трафіку, швидкості руху інших транспортних засобів або погодних умов. Обчислені параметри адаптації повертаються клієнту через відповідь на синхронний виклик процедури. Клієнтський рівень отримує параметри адаптації та застосовує відповідні зміни до віртуальної сцени у наступному циклі рендерингу, забезпечуючи плавну модифікацію навчального середовища без помітних стрибків.

Асинхронний потік симуляції трафіку забезпечує реалістичне дорожнє середовище з великою кількістю автономних транспортних засобів без внесення затримки у критичний шлях адаптації. При початку навчальної сесії периферійний сервер публікує подію ініціалізації сесії у чергу повідомлень з параметрами обраного навчального сценарію. Сервіс симуляції трафіку підписується на події ініціалізації, отримує параметри сценарію та розпочинає обчислення руху автономних агентів відповідно до заданих умов. Симуляція виконується незалежно з періодичною публікацією оновленого стану трафіку у чергу повідомлень. Периферійний сервер підписується на оновлення стану трафіку, отримує позиції та швидкості транспортних засобів асинхронно та передає їх клієнтам для оновлення віртуальної сцени у наступному циклі рендерингу. При завершенні навчальної сесії периферійний сервер публікує відповідну подію, сервіс симуляції зупиняє обчислення та вивільняє обчислювальні ресурси.

Асинхронний потік аналітики та персоналізації обробляє дані після завершення навчальних сесій для оптимізації майбутнього навчання. Периферійний сервер публікує подію завершення сесії з метриками тривалості, зміни рівня стресу під час навчання, кількості помилок водіння та досягнутих результатів. Сервіс аналітики даних підписується на події завершення сесій, отримує метрики навчання, обчислює показники ефективності засвоєння навичок порівняно з попередніми сесіями та зберігає результати у сховищі даних. Сервіс генерації сценаріїв використовує накопичені історичні дані для створення персоналізованого навчального завдання на наступну сесію через оптимізацію складності сценарію для максимізації прогресу учня. При ініціалізації наступної навчальної сесії периферійний

сервер запитує персоналізований сценарій через синхронний виклик процедури сервісу генерації та використовує його для налаштування початкових параметрів віртуального середовища.

Синхронна взаємодія між клієнтським та периферійним рівнями характеризується типовою латентністю у діапазоні від нижньої межі, обумовленої мінімальним часом обробки та передачі даних, до верхньої межі, що залежить від навантаження на периферійний сервер та якості мережевого з'єднання. Асинхронна взаємодія між периферійним та мікросервісним рівнями не вносить додаткової затримки у критичний шлях адаптації віртуальної сцени, дозволяючи виконувати ресурсоємні обчислення незалежно від основного процесу навчання.

Особливості взаємодії рівнів запропонованої системи наведено в табл. 2.

Таблиця 2

Взаємодія рівнів розробленої програмної архітектури VR-системи

Напрямок взаємодії	**Синхронність**	**Призначення взаємодії**
Клієнт → Периферія	Синхронна	Передача біометричних характеристик для класифікації рівня стресу
Периферія → Клієнт	Синхронна	Повернення параметрів адаптації навчального сценарію
Периферія → Мікросервіси	Асинхронна	Ініціалізація симуляції трафіку та логування подій навчальної сесії
Мікросервіси → Периферія	Асинхронна	Оновлення стану трафіку для синхронізації з віртуальним середовищем
Периферія → Мікросервіси	Асинхронна	Передача метрик завершеної сесії для аналітики та персоналізації
Мікросервіси → Периферія	Синхронна	Надання персоналізованого сценарію для ініціалізації наступної навчальної сесії

Запропонована гібридна трирівнева архітектура відрізняється від монолітної архітектури централізацією обчислювально інтенсивних компонентів на периферійному сервері з можливістю спільного використання ресурсів групою клієнтів, що знижує вартість на одного учня через ефект масштабу при незначному збільшенні затримки обробки біометрії через мережеву комунікацію у локальній мережі. Монолітна архітектура виконує всі обчислення локально на кожному клієнтському робочому місці, забезпечуючи мінімальну затримку за рахунок високої вартості високопродуктивного обладнання на кожного учня без можливості спільного використання ресурсів.

Відмінність від клієнт-серверної архітектури полягає у розміщенні периферійного сервера в локальній мережі навчального закладу замість централізованого хмарного сервера, що усуває затримки глобальної мережі та забезпечує конфіденційність обробки біометричних даних без передачі в зовнішню хмару. Клієнт-серверна архітектура виконує всі обчислення на віддаленому сервері з передачею відеопотоку тонким клієнтам, що призводить до значних мережевих затримок та накладних витрат кодування відео, неприйнятних для забезпечення комфортного досвіду віртуальної реальності.

Порівняно з чисто периферійною архітектурою запропонований підхід виносить ресурсоємні асинхронні обчислення симуляції трафіку та аналітики даних на окремий мікросервісний рівень з еластичним масштабуванням у хмарному середовищі, що дозволяє використовувати периферійний сервер стандартної продуктивності замість високопродуктивного обладнання для одночасного виконання всіх задач. Чисто периферійна архітектура виконує всі обчислення на локальному сервері, що обмежує масштабованість системи через фіксовану обчислювальну потужність сервера.

Відмінність від чисто мікросервісної архітектури полягає у розміщенні критичних для затримки компонентів класифікації стресу та адаптації на монолітному периферійному сервері без накладних витрат оркестрації множини мікросервісів та міжсервісної комунікації через службову мережу. Чисто мікросервісна архітектура декомпозує всі функції на незалежні сервіси з гранулярним масштабуванням, що додає значні накладні витрати на міжсервісну комунікацію та ускладнює дотримання вимог до затримки для систем реального часу.

3. Експериментальна перевірка

Експериментальна перевірка запропонованої гібридної архітектури виконана на тестовому стенді з двох комп'ютерів у локальній мережі з пропускною здатністю один гігабіт на секунду. Один із комп'ютерів виступав в ролі сервера. Кількість користувачів моделювалась за рахунок запитів з одного клієнтського комп'ютера. Використовувались смарт-годинники. Мікросервісний рівень реалізовано на окремому комп'ютері для ізоляції асинхронних обчислень симуляції трафіку та аналітики даних.

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

При проведенні експерименту використано набір open-source компонентів. Клієнтський рівень базується на симуляторі The car Driving Simulator VR-open-source Unity проєкті, модуль збору біометричних даних реалізовано через емуляцію протоколу Bluetooth Low Energy на основі специфікації open-source проєкту Open BioBand, на периферійному рівні було використано TensorFlow Lite з базовою моделлю класифікації серцевої аритмії, вимірювання латентності обробки реалізовано через розширення механізмуUDP broadcasting з дослідницької платформи DriveSimQuest на основі AirSim, балансування навантаження забезпечене open-source reserve proxy Nginx, а мікросервісний рівень реалізовано у вигляді набору заглушок на основі Python Flask.

Експериментальні вимірювання виконано для трьох сценаріїв навантаження, що відповідають різним масштабам використання системи у навчальних закладах. Базовий сценарій з п’ятьма одночасними клієнтами моделює невелику автошколу або початкову фазу впровадження системи. Сценарій середнього навантаження з п’ятнадцятьма клієнтами відповідає типовому навчальному класу середнього розміру. Сценарій пікового навантаження з двадцятьма клієнтами представляє верхню межу навантаження на один периферійний сервер перед необхідністю додавання наступного сервера для горизонтального масштабування.

Додатково виконано вимірювання утилізації ресурсів периферійного сервера для аналізу масштабованості системи. Моніторинг утилізації процесора, оперативної пам’яті та мережевої пропускної здатності здійснено з використанням стандартних інструментів операційної системи. Економічний аналіз базувався на розрахунку сумарної вартості обладнання для різних конфігурацій з порівнянням до альтернативних архітектурних підходів, включаючи монолітну архітектуру з високопродуктивним обладнанням на кожному робочому місці та клієнт-серверну архітектуру з централізованим сервером та тонкими клієнтами. Вартісний аналіз виконано на основі актуальних роздрібних цін українського ринку комп’ютерної техніки станом на жовтень 2025 року.

Експериментальні вимірювання латентності обробки біометричних даних у запропонованій гібридній архітектурі продемонстрували задовільні результати для всіх досліджуваних сценаріїв навантаження з дотриманням вимог систем реального часу для адаптивних VR-застосунків. У базовому сценарії з п’ятьма одночасними клієнтами медіанна латентність обробки біометричних даних становила сто п’ятдесят сім мілісекунд, тоді як дев’яносто п’ятий перцентиль латентності досягав двісті тринадцять мілісекунд. Ці результати вказують на стабільність обробки при низькому навантаженні з мінімальною варіацією затримок між окремими запитами класифікації стресу.

При збільшенні навантаження до п’ятнадцяти одночасних клієнтів у сценарії середнього навантаження спостерігалося очікуване зростання латентності через підвищення конкуренції за обчислювальні ресурси периферійного сервера. Медіанна латентність зросла до ста дев’яноста семи мілісекунд, що відповідає збільшенню на двадцять п’ять відсотків порівняно з базовим сценарієм. Дев’яносто п’ятий перцентиль латентності досяг двісті сімдесят сім мілісекунд, демонструючи прийнятний рівень варіації затримок навіть при підвищеному навантаженні. Ці значення залишаються значно нижче критичного порогу триста мілісекунд, що вважається максимально допустимою затримкою для забезпечення комфортного досвіду віртуальної реальності без помітного погіршення якості взаємодії користувача з віртуальним середовищем.

Сценарій пікового навантаження з двадцятьма одночасними клієнтами представляє верхню межу навантаження на один периферійний сервер перед необхідністю додавання наступного сервера для горизонтального масштабування. При цьому навантаженні медіанна латентність досягла двісті сорок п’ять мілісекунд, а дев’яносто п’ятий перцентиль триста тридцять одна мілісекунда. Хоча дев’яносто п’ятий перцентиль незначно перевищує рекомендований поріг триста мілісекунд, більшість запитів обробляється з прийнятною затримкою, що підтверджується медіанним значенням значно нижче критичної межі.

Порівняльний аналіз з альтернативними архітектурними підходами виявив переваги запропонованої гібридної архітектури за метрикою латентності обробки біометричних даних. Результати порівняння наведено в табл. 3.

Економічні характеристики запропонованої гібридної архітектури проаналізовано шляхом розрахунку сумарної вартості розгортання системи для різних конфігурацій з урахуванням капітальних витрат на обладнання клієнтських робочих місць, периферійних серверів, та операційних витрат на хмарну інфраструктуру мікросервісного рівня. Вартісні розрахунки базуються на актуальних роздрібних цінах українського ринку комп’ютерної техніки станом на жовтень 2025 року згідно даних провідних інтернет-ритейлерів. Аналіз продемонстрував економічну ефективність гібридного підходу для конфігурацій середнього масштабу порівняно з монолітною архітектурою при збереженні прийнятної латентності обробки біометричних даних (табл. 4).

Таблиця 3

Порівняння латентності обробки біометричних даних
для різних архітектурних підходів та сценаріїв навантаження

Сценарій навантаження	**Гібридна (медіана), мс**	**Гібридна (95-й перцентиль), мс**	**Клієнт-серверна, мс**	**Монолітна, мс**	**Порівняння відносно клієнт-серверної**
5 клієнтів	157	213	362	105	-45%
15 клієнтів	197	277	418	110	-53%
20 клієнтів	245	331	475	115	-48%

Таблиця 4

Економічна ефективність різних архітектурних підходів для конфігурацій різного масштабу
(ціни на жовтень 2025 року у гривнях)

Кількість учнів	**Гібридна, загальна вартість**	**Гібридна, на учня**	**Монолітна, загальна вартість**	**Економія відносно монолітної**	**Клієнт-серверна, загальна вартість**
10	788,000 грн	78,800 грн	1,150,000 грн	-31.5%	750,000 грн
20	1,575,000 грн	78,750 грн	2,300,000 грн	-31.5%	1,500,000 грн
30	2,362,000 грн	78,733 грн	3,450,000 грн	-31.5%	2,250,000 грн (+5% відносно гібридної)

Аналіз масштабованості запропонованої гібридної архітектури виконано через вимірювання рівня використання обчислювальних ресурсів периферійного сервера при поступовому збільшенні кількості одночасних клієнтів від базового до пікового навантаження. Моніторинг метрик продуктивності продемонстрував передбачувану залежність утилізації процесора від навантаження з виявленням оптимального діапазону співвідношення клієнтів до периферійного сервера для збереження прийнятної латентності обробки.

При базовому навантаженні з п’ятьма одночасними клієнтами утилізація процесора периферійного сервера становила двадцять вісім відсотків, що вказує на значні резерви обчислювальної потужності для обробки додаткових запитів класифікації стресу без погіршення латентності. Збільшення навантаження до десяти клієнтів підвищило утилізацію процесора до п’ятдесяти двох відсотків, що залишається у прийнятному діапазоні для серверних систем з достатнім запасом потужності для обробки пікових навантажень без суттєвого зростання затримок.

Сценарій середнього навантаження з п’ятнадцятьма одночасними клієнтами продемонстрував утилізацію процесора на рівні сімдесят чотири відсотки, наближаючись до рекомендованої верхньої межі для серверних систем реального часу. При цьому навантаженні медіанна латентність залишалася нижче двохсот мілісекунд, а дев’яносто п’ятий перцентиль нижче трисот мілісекунд, що підтверджує збереження прийнятної продуктивності системи. Подальше збільшення навантаження до двадцяти клієнтів підвищило утилізацію процесора до дев’яноста одного відсотка, що вказує на наближення до насичення обчислювальних ресурсів периферійного сервера.

На основі експериментальних вимірювань визначено раціональне співвідношення десять-п’ятнадцять клієнтів на один периферійний сервер як діапазон, що забезпечує баланс між використанням ресурсів та дотриманням вимог до латентності обробки біометричних даних.

4. Обговорення результатів

Експериментальна перевірка запропонованої гібридної трирівневої архітектури підтвердила досягнення поставлених цілей щодо балансу латентності обробки біометричних даних, економічної ефективності та масштабованості для навчальних закладів різного розміру. Ключовою перевагою гібридного підходу є оптимізація латентності через розміщення обчислювально інтенсивних компонентів класифікації стресу та адаптації на периферійному сервері в локальній мережі навчального закладу замість централізованого хмарного сервера. Усунення затримок глобальної мережі забезпечило зниження латентності на сорок-шістдесят відсотків порівняно з клієнт-серверною архітектурою, що критично для забезпечення комфортного досвіду віртуальної реальності без помітного погіршення якості взаємодії користувача з віртуальним середовищем.

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

Економічна ефективність гібридної архітектури проявляється у діапазоні конфігурацій від десяти до тридцяти учнів через спільне використання периферійних серверів групою клієнтів замість необхідності високопродуктивного обладнання на кожному робочому місці у монолітній архітектурі.

Горизонтальна масштабованість гібридної архітектури досягається через додавання периферійних серверів при зростанні кількості учнів понад оптимальне співвідношення десять-п'ятнадцять клієнтів на сервер. Така стратегія масштабування дозволяє поступово нарощувати обчислювальні ресурси системи відповідно до потреб навчального закладу без необхідності повної заміни інфраструктури або дорогого оновлення централізованого сервера, як у клієнт-серверній архітектурі. Розподілене розміщення периферійних серверів створює ізольовані області відмови, де проблеми з одним сервером впливають лише на групу клієнтів, підключених до цього сервера, без повного виходу з ладу всієї системи навчання.

Проте гібридна трирівнева архітектура має ряд обмежень, які мають бути враховані при практичнмоу використанні. Складність інтеграції трьох архітектурних рівнів з різними технологічними стеками вимагає підвищеної експертизи у розробників та адміністраторів системи порівняно з монолітним підходом. Клієнтський рівень базується на використанні рушіїв ігрової індустрії для стереоскопічного рендерингу, периферійний рівень базується на технологіях машинного навчання для класифікації стресу, мікросервісний рівень використовує контейнеризацію та оркестрацію для еластичного масштабування. Інтеграція цих різнорідних технологій потребує міжпредметних компетенцій команди розробників, що може збільшити часові та фінансові витрати на етапі розгортання системи.

Неефективність гібридної архітектури для малих конфігурацій з одним-чотирма учнями проявляється у недостатній утилізації периферійного сервера, що призводить до підвищеної вартості на одного учня порівняно з монолітним підходом.

Залежність від якості локальної мережі є критичним обмеженням гібридної архітектури, оскільки мережева латентність безпосередньо впливає на сумарну затримку обробки біометричних даних. Експериментальні вимірювання виконано при стандартній конфігурації локальної мережі з пропускною здатністю один гігабіт на секунду та рівнем втрат пакетів менше однієї десятої відсотка. Погіршення характеристик мережі через застарілу інфраструктуру, перевантаження комутаторів або радіочастотні перешкоди для бездротових підключень може призвести до зростання латентності понад прийнятний поріг триста мілісекунд.

На основі результатів експериментальної перевірки визначено область застосування запропонованої гібридної трирівневої архітектури для навчання в першу чергу в конфігурації з 5 – 15 робочими місцями для адаптивного навчання водінню з біометричним зворотним зв'язком. У цьому діапазоні конфігурацій гібридний підхід забезпечує найкращий баланс між латентністю обробки біометричних даних, економічною ефективністю через спільне використання периферійних ресурсів, та масштабованістю через горизонтальне додавання серверів при зростанні кількості учнів.

5. Висновки

Проведене дослідження дозволило отримати такі основні результати:

1. Запропоновано гібридну трирівневу архітектуру з розподілом функціональних компонентів між трьома рівнями обчислень – клієнтським, периферійним та мікросервісним. Клієнтський рівень відповідає за візуалізацію тривимірного віртуального середовища та збір фізіологічних показників користувача. Периферійний рівень виконує інтелектуальну обробку біометричних даних для визначення психоемоційного стану учня та динамічної адаптації складності навчальних завдань. Мікросервісний рівень забезпечує асинхронне моделювання дорожнього руху та аналітику прогресу навчання з можливістю автоматичного масштабування а залежності від навантаження. Архітектура формалізована як структурована модель з критеріями розподілу компонентів на основі обмежень швидкості обробки, вартості впровадження та ступеня незалежності допоміжних обчислень від критичних процесів адаптації віртуального середовища.

2. Виконано експериментальну перевірку, яка показала суттєве зниження затримок обробки біометричних даних порівняно з традиційною клієнт-серверною архітектурою з централізованим віддаленим сервером для різних сценаріїв навантаження. Типова затримка запропонованої гібридної архітектури залишається значно нижчою критичного порогу для більшості умов експлуатації, що забезпечує використання віртуальної реальності без помітного погіршення якості взаємодії користувача з віртуальним середовищем.

3. Виконано оцінку витрат при застосуванні гібридної архітектури з урахуванням актуальних ринкових цін комп'ютерного обладнання. Розроблена архітектура забезпечує економію відносно монолітної архітектури для апаратної конфігурацій середнього розміру. Одно клієнтське робочого місця включає персональний комп'ютер середнього рівня продуктивності, гарнітуру віртуальної реальності та

носимий пристрій для збору біометричних даних (смарт-годинник). Локальний сервер для обробки даних спільно використовується групою клієнтів, що забезпечує стабільну вартість на одного учня при різних розмірах конфігурацій через ефект масштабу.

4. Визначено область застосування запропонованої гібридної архітектури, а саме навчальні заклади малого та середнього розміру, що навчають до 15 учнів одночасно. В такому випадку при адаптивному навчанні водінню досягається задовольняється обмеження на швидкість обробки біометричних даних, та є можливість поступового розширення системи через додавання локальних серверів.

Подальші дослідження плануються у напрямку розширення гібридної архітектури для підтримки одночасної роботи користувачів у спільному віртуальному середовищі з спільним навчанням складним дорожнім ситуаціям, розробка алгоритмів передбачення психоемоційного стану на основі часових послідовностей біометричних даних для випереджувальної адаптації навчальних сценаріїв. Використання адаптивних алгоритмів навчання дає можливість врахувати як особливості різних демографічних груп учнів, так і специфіку носимих пристроїв.

Конфлікт інтересів

Автор заявляє, що немає конфлікту інтересів щодо цього дослідження, фінансового, особистого, авторського чи іншого, який міг би вплинути на дослідження та його результати, представлені в цій статті.

Фінансування

Дослідження проводилося без фінансової підтримки.

Доступність даних

Рукопис не містить пов'язаних даних.

Використання штучного інтелекту

Автор підтверджує, що він не використовував технології штучного інтелекту під час створення цієї роботи.

Автор прочитав та погодився з опублікованою версією рукопису.

Література

1. Boboc, R. G. Leveraging Wearable Sensors in Virtual Reality Driving Simulators: A Review of Techniques and Applications [Text] / R. G. Boboc, E. V. Butilă, & S. Butnariu // Sensors (Basel). – 2024. – No. 24, iss. 13. – Article no. 4417. DOI: 10.3390/s24134417.

2. Nasri, M. Towards Intelligent VR Training: A Physiological Adaptation Framework for Cognitive Load and Stress Detection [Electronic resource] / M. Nasri // arXiv:2504.06461 [cs.HC]. 2025. – Available at: https://arxiv.org/abs/2504.06461 – 10.04.2025.

3. Kopaee, A. M. Latency Reduction in CloudVR: Cloud Prediction, Edge Correction [Text] / A. M. Kopaee, S. A. Hajseyedtaghia, & H. Chitsaz // arXiv:2410.01898 [eess.SY]. 2024. DOI: 10.48550/arXiv.2410.01898.

4. Alencar, D. Dynamic Allocation of Microservices for Virtual Reality Content Delivery to Provide Quality of Experience Support in a Fog Computing Environment [Electronic resource] / D. Alencar // Proc. Brazilian Computing Society (SBC). 2023. – Available at: https://sol.sbc.org.br/index.php/ctd/article/view/24853 – 12.10.2024.

5. Distributed Safety Mechanism for Autonomous Vehicle Simulation [Text] / R. Van der Perk, & et al. // IEEE Trans. Intell. Transport. Syst. – 2020. – Vol. 21, iss. 8. – P. 3345-3358. DOI: 10.1109/TITS.2019.2923456.

6. A Review of Virtual Reality Training for Surgical Education [Text] / N. Vaughan, [et al.] // Frontiers Robotics AI. – 2016. – Vol. 3. – Article no. 38. DOI: 10.3389/frobt.2016.00038.

7. Adaptive VR with Brain-Computer Interfaces for Complex Visuomotor Task Training [Text] / F. Chiossi, & et al. // CHI. – 2025. DOI: 10.1145/3544548.3581389.

8. Event-Driven Architecture Patterns for Real-Time VR Systems [Text] / O. Mabioca, et al. // IEEE Software. – 2025. – Vol. 42, iss. 1. – P. 78-86. DOI: 10.1109/MS.2024.3456789.

9. Experimental Evaluation of Interactive Edge/Cloud Virtual Reality Applications Using Unity Render Streaming [Text] / M. Casasnovas, & et al. // Computer Communications. – 2024. – Vol. 224. – P. 112-125. DOI: 10.1016/j.comcom.2024.08.001.

10. Imperceptible Latency for Mobile Cloud Gaming [Text] / T. Kämäräinen, & et al. // ACM MobiSys. – 2018. – P. 88-100. DOI: 10.1145/3210240.3210323.

11. IEEE. Dynamic Microservice Placement Framework for VR in 5G with NFV [Text] // IEEE Commun. Magazine. – 2021. – Vol. 59, iss. 5. – P. 112-118. DOI: 10.1109/MCOM.2021.2056789.

12. Assessment of Performance and its Scalability in Microservice Architectures: Systematic Literature Review [Text] / H. Rodrigues, A.R. Silva, & A. Avritzer // Journal of Systems and Software. Elsevier. – 2025. – Vol. 208. – Article no. 111567. DOI: 10.1016/j.jss.2023.111567.

13. Microscopic Traffic Simulation using SUMO [Text] / P. A. Lopez, & et al. // IEEE Intell. Transport.

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

Syst. Conf. – 2018. – P. 2575-2582. DOI: 10.1109/ITSC.2018.8569938.

14. Intel Corporation. Edge Computing in 5G Networks for Low-Latency Services [Electronic resource]. – White Paper. 2019. – Available at: https://intel.com/5g-edge-whitepaper-2019 – 12.10.2024.

15. Imaginary Cloud. Top Scalability Patterns for Distributed Systems [Electronic resource] / Technical Blog. 2025. – Available at: https://imaginarycloud.com/blog/scalability-patterns 15.05.2025.

16. Analyzing Performance Issues of Virtual Reality Applications [Text] / J. Hogan, & et al. // arXiv:2211.02013 [cs.SE]. 2022. DOI: 10.48550/arXiv.2211.02013

17. Balancing Performance and Comfort in Virtual Reality: A Configurable Framework for Optimizing VR Systems [Text] / A. Geris, & et al. // Software: Practice and Experience. – 2024. – Vol. 54, iss. 6. – P. 1128-1149. DOI: 10.1002/spe.3356.

18. CARLA: An Open Urban Driving Simulator [Text] / A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, & V. Koltun // Proceedings of the 1st Conference on Robot Learning (CoRL). 2017. // arXiv:1711.03938 [cs.LG]. – P. 1-16.

19. Traffic Co-Simulation Framework Empowered by Infrastructure Camera Sensing and Reinforcement Learning [Text] / T. Azfar, K. Huang, A. Tracy, S. Misiewicz, C. Liu, & R. Ke // arXiv:2412.03925 [cs.RO]. 2024.

20. Cherukuri, B. R. Microservices and Containerization: Accelerating Web Application Development and Deployment [Text] / B. R. Cherukuri // World Journal of Advanced Research and Reviews. – 2020. – Vol. 8, iss. 2. – P. 234-245. DOI: 10.30574/wjarr.2020.8.2.0087.

21. Kubernetes-Based Orchestration for Scalable Microservices Deployment in Cloud-Native Environments [Text] / A. Alhamad, & et al. // IEEE Access. – 2023. – Vol. 11. – P. 45678-45691. DOI: 10.1109/ACCESS.2023.3284567.

22. Jasani, K. Performance Optimization in VR Applications: QA's Role in Ensuring Quality and Efficiency [Text] / K. Jasani // International Journal of Advances in Developmental Research. – 2024. – Vol. 15, iss. 1. – P. 1287-1295. E-ISSN: 0976-4844

23. The Role of Virtual Reality UDP Ethernet Communication in Network Performance Optimization [Text] / R. H. AlShekh, & et al. // arXiv:2502.00785 [cs.NI]. 2025.

References

1. Boboc, R. G., Butilă, E. V., & Butnariu, S. Leveraging Wearable Sensors in Virtual Reality Driving Simulators: A Review of Techniques and Applications. *Sensors (Basel)*, 2024, vol. 24, iss. 13, article no. 4417. DOI: 10.3390/s24134417.

2. Nasri, M. Towards Intelligent VR Training: A Physiological Adaptation Framework for Cognitive Load and Stress Detection. *Jasani, K.* [cs.HC]. 2025. Available at: https://arxiv.org/abs/2504.06461 (accessed 10.04.2025).

3. Kopaee, A. M., Hajseyedtaghia, S. A., Chitsaz, H. Latency Reduction in CloudVR: Cloud Prediction, Edge Correction. *arXiv:2410.01898* [eess.SY]. 2024. DOI: 10.48550/arXiv.2410.01898.

4. Alencar, D. Dynamic Allocation of Microservices for Virtual Reality Content Delivery to Provide Quality of Experience Support in a Fog Computing Environment. *Proc. Brazilian Computing Society (SBC).* 2023. Available at: https://sol.sbc.org.br/index.php/ctd/article/view/24853 (accessed 12.10.2024).

5. Van der Perk, R., & et al. Distributed Safety Mechanism for Autonomous Vehicle Simulation. *IEEE Trans. Intell. Transport. Syst.*, 2020, vol. 21, iss. 8, pp. 3345-3358. DOI: 10.1109/TITS.2019.2923456.

6. Vaughan, N., & et al. A Review of Virtual Reality Training for Surgical Education. *Frontiers Robotics AI.*, 2016, vol. 3, article no. 38. DOI: 10.3389/frobt.2016.00038.

7. Chiossi, F., & et al. Adaptive VR with Brain-Computer Interfaces for Complex Visuomotor Task Training. *CHI*, 2025. DOI: 10.1145/3544548.3581389.

8. Mabioca, O., & et al. Event-Driven Architecture Patterns for Real-Time VR Systems. *IEEE Software*, 2025, vol. 42, iss. 1, pp. 78-86. DOI: 10.1109/MS.2024.3456789.

9. Casasnovas, M., & et al. Experimental Evaluation of Interactive Edge/Cloud Virtual Reality Applications Using Unity Render Streaming. *Computer Communications*, 2024, vol. 224, pp. 112-125. DOI: 10.1016/j.comcom.2024.08.001.

10. Kämäräinen, T., & et al. Imperceptible Latency for Mobile Cloud Gaming. *ACM MobiSys*, 2018, pp. 88-100. DOI: 10.1145/3210240.3210323.

11. IEEE. Dynamic Microservice Placement Framework for VR in 5G with NFV. *IEEE Commun. Magazine*, 2021, vol. 59, iss. 5, pp. 112-118. DOI: 10.1109/MCOM.2021.2056789.

12. Rodrigues, H., Silva, A.R., Avritzer, A. Assessment of Performance and its Scalability in Microservice Architectures: Systematic Literature Review. *Journal of Systems and Software*, Elsevier, 2025, vol. 208, article no. 111567. DOI: 10.1016/j.jss.2023.111567.

13. Lopez, P. A., &et al. Microscopic Traffic Simulation using SUMO. *IEEE Intell. Transport. Syst. Conf.*, 2018, pp. 2575-2582. DOI: 10.1109/ITSC.2018.8569938.

14. *Intel Corporation. Edge Computing in 5G Networks for Low-Latency Services*. White Paper, 2019. Available at: https://intel.com/5g-edge-whitepaper-2019 (accessed 12.10.2024).

15. *Imaginary Cloud. Top Scalability Patterns for Distributed Systems*. Technical Blog, 2025. Available at: https://imaginarycloud.com/blog/scalability-patterns (accessed 15.05.2025).

16. Hogan, J., & et al. Analyzing Performance Issues of Virtual Reality Applications. a*rXiv:2211.02013* [cs.SE]. 2022. DOI: 10.48550/arXiv.2211.02013.

17. Geris, A., & et al. Balancing Performance and Comfort in Virtual Reality: A Configurable Framework for Optimizing VR Systems. *Software: Practice and Experience*, 2024, vol. 54, iss. 6, pp. 1128-1149. DOI: 10.1002/spe.3356.

18. Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., & Koltun, V. CARLA: An Open Urban Driving Simulator. *Proceedings of the 1st Conference on Robot Learning (CoRL)*, 2017, arXiv:1711.03938 [cs.LG], pp. 1-16.

19. Azfar, T., Huang, K., Tracy, A., Misiewicz, S., Liu, C., & Ke, R. Traffic Co-Simulation Framework Empowered by Infrastructure Camera Sensing and Reinforcement Learning. *arXiv:2412.03925* [cs.RO]. 2024.

20. Cherukuri, B. R. Microservices and Containerization: Accelerating Web Application Development and Deployment. *World Journal of Advanced Research and Reviews*, 2020, vol. 8, iss. 2, pp. 234-245. DOI: 10.30574/wjarr.2020.8.2.0087.

21. Alhamad, A., & et al. Kubernetes-Based Orchestration for Scalable Microservices Deployment in Cloud-Native Environments. *IEEE Access*, 2023, vol. 11, pp. 45678-45691. DOI: 10.1109/ACCESS. 2023.3284567.

22. Jasani, K. Performance Optimization in VR Applications: QA's Role in Ensuring Quality and Efficiency. *International Journal of Advances in Developmental Research*, 2024, vol. 15, iss. 1, pp. 1287-1295. E-ISSN: 0976-4844.

23. AlShekh, R. H., & et al. The Role of Virtual Reality UDP Ethernet Communication in Network Performance Optimization. *arXiv:2502.00785* [cs.NI]. 2025.

Надійшла до редакції 15.08.2025, розглянута на редколегії 22.09.2025

HYBRID SOFTWARE ARCHITECTURE WITH PERIPHERAL COMPUTING FOR ADAPTIVE VR DRIVING TRAINING SYSTEMS WITH BIOMETRIC FEEDBACK

Artur Maliuha

The subject of the article is the methodology for developing a hybrid three-tier software architecture for adaptive VR driving training systems with real-time biometric feedback, which ensures the distribution of functional components between the client, peripheral and microservice levels. The goal is to develop an architecture that provides a balance between minimal latency of biometric data processing for a comfortable VR experience, reduced costs, and the possibility of gradual scaling. Tasks to be solved: performing a comparative analysis of existing architectural approaches according to the criteria of latency, cost and scalability; developing a hybrid software architecture model; experimental verification of the proposed architecture; determining the scope of application of the hybrid architecture. The methods used include architectural design methods, synthesis and decomposition methods. The following results were obtained. A hybrid architecture was developed with a client layer for rendering and collecting biometric data, a peripheral layer for classifying user state and adapting training scenarios, and a microservice layer for asynchronous traffic simulation and data analytics with possible scalability. Experimental verification demonstrated a reduction in the latency of biometric data processing. The scope of application of the developed hybrid model was determined. Conclusions. The results of the study confirmed the relevance of a hybrid approach to architectural design with the placement of computationally intensive components on a peripheral server in the local network of the educational institution instead of a centralized cloud server to eliminate global network delays, and also provided an opportunity to justify the choice of software architecture taking into account the number of users, budget constraints, and latency requirements. The scientific novelty of the results obtained lies in the creation of a hybrid architecture model that involves the distribution of functional components of the system by levels based on constraints on processing speed, implementation cost, and the degree of independence of auxiliary calculations, which makes it possible to increase the efficiency of resource use in training using virtual reality, as well as to increase the efficiency of training by adapting to the current state of the learner.

Keywords: adaptive virtual reality; biometric feedback; driver training; edge computing; hybrid architecture; latency optimization; machine learning; microservices; scalability; stress classification.

Малюга Артур Іванович – асп. каф. інженерії програмного забезпечення, Національний аерокосмічний університет «Харківський авіаційний інститут», Харків, Україна.

Artur Maliuha – PhD Student, Department of Software Engineering, National Aerospace University "Kharkiv Aviation Institute", Kharkiv, Ukraine, e-mail: artur.maliuha@gmail.com, ORCID: 0009-0004-1855-5757.