*DOI: https://doi.org/10.25140/2411-5363-2025-3(41)-225-236*
*УДК 004.94:681.5*

***Петро Дмитрович Олексієнко[1], Володимир Вікторович Казимир[2]***

[1]аспірант, здобувач наукового ступеня доктор філософії за спеціальністю 122
Інститут проблем математичних машин і систем Національної академії наук України (Київ, Україна)
**E-mail:** alexeenkop77@gmail.com. **ORCID:** https://orcid.org/0009-0009-9588-9490

[2]доктор технічних наук, професор, професор кафедри інформаційних та комп'ютерних систем
Національний університет «Чернігівська політехніка» (Чернігів, Україна)
**E-mail:** vvkazymyr@gmail.com. **ORCID:** https://orcid.org/0000-0001-8163-1119. **ResearcherID:** Q-2925-2016

# ПОДІЄВО-ОРІЄНТОВАНИЙ ПІДХІД У РЕАЛІЗАЦІЇ ВБУДОВАНИХ СИСТЕМ КЕРУВАННЯ РОЄМ БПЛА

*У статті представлено результати експериментального дослідження подієво-орієнтованого підходу в реалізації вбудованих систем керування роєм БПЛА на основі керуючих Е-мереж (Control E-Networks). Перевірка ефективності проводиться шляхом порівняння запропонованої схеми керування на основі слухача подій із схемою циклічного опитування у серії експериментів: базове порівняння, багатоточковий аналіз, оцінка стабільності та аналіз алгоритмічної складності. Результати проведеного дослідження підтвердили суттєву перевагу подієво-орієнтованого підходу за основними цільовими та функціональними критеріями рою БПЛА: швидкістю реакції, стабільністю та ефективністю використання ресурсів.*

***Ключові слова:*** *рої БПЛА; керуючі Е-мережі; системи керування; подієво-орієнтоване керування; циклічне керування.*
*Рис.: 6. Бібл.: 12.*

**Актуальність теми дослідження**. Сучасні мультиагентні системи, зокрема рої безпілотних літальних апаратів (БПЛА), вимагають швидкої, стабільної та енергоефективної обробки подій у режимі реального часу. Традиційні підходи до організації управління у вбудованих системах ґрунтуються на циклічному опитуванні, що забезпечує контроль за станом системи, але створює надмірне навантаження на процесор, знижує передбачуваність поведінки та обмежує масштабованість зі зростанням кількості агентів. Це суперечить сучасним тенденціям розвитку автономних технологій, де ключовими стають вимоги до безпеки, передбачуваності реакцій та мінімізації споживання енергії.

У цьому контексті актуальним є застосування подієво-орієнтованих підходів, які демонструють значні переваги у суміжних галузях – від подієво-тригерного керування в багатодронових системах [1] до використання подієвих датчиків у системах технічного зору [2]. Для галузі управління роями БПЛА така трансформація архітектури є критично важливою, оскільки дозволяє одночасно підвищити швидкість реакції, покращити надійність системи та знизити енергетичні витрати під час виконання довготривалих автономних місій.

**Постановка проблеми.** Попри значний прогрес у розвитку подієво-орієнтованих підходів, у практиці вбудованих систем керування досі переважає циклічний підхід. Він є простим у реалізації, однак має низку критичних недоліків. По-перше, постійне опитування призводить до значного навантаження на центральний процесор навіть у моменти, коли подій немає, що особливо проблематично для енергообмежених платформ БПЛА. По-друге, затримка реакції залежить від параметрів циклу та навантаження системи, що робить її непередбачуваною у критичних сценаріях, таких як уникнення зіткнень або аварійна посадка. По-третє, із зростанням кількості дронів у роєвій місії масштабованість системи деградує, оскільки складність зростає лінійно від часу очікування та кількості агентів [3].

Таким чином, існує проблема пошуку рішення, яке б поєднувало швидкість реакції, передбачуваність, ефективність використання ресурсів та здатність до масштабування. Подієво-орієнтований підхід на основі механізму слухача подій потенційно вирішує цю проблему, але потребує ґрунтовної кількісної перевірки на основі формальних моделей та експериментальних даних.

**Аналіз останніх досліджень і публікацій показує,** що за останні роки подієво-тригерне керування (event-triggered control – ETC) перейшло з вузьких застосувань у зрілу методологію ресурсоефективного керування. Оглядова стаття [4] систематизує сучасні

підходи до формування тригерів, проблематику запобігання явищу «Zeno» (накопичення нескінченної кількості спрацювань за скінченний час через зменшення міжподієвих інтервалів до нуля), спільне проєктування керування та планування передавання даних, а також підкреслює переваги ETC щодо скорочення частоти обчислень і обміну повідомленнями без втрати якості керування. Окремий огляд динамічних тригерів [5] детально розбирає схеми зі змінним порогом, що дозволяють відокремити частоту дискретизації вимірювань від частоти оновлення керуючих дій, підтримуючи гарантовану стабільність та зменшуючи кількість обчислень спостерігача/регулятора. Для багатоагентних систем релевантний огляд [6] показує, що подієві правила консенсусу знижують навантаження на мережу та процесор, водночас зберігаючи збіжність і стійкість до збурень – це безпосередньо важливо для роєвих сценаріїв.

У застосуванні до безпілотних літальних апаратів низка робіт демонструє практичні вигоди ETC на рівні задач місії. У [7] запропоновано подієво-тригерний контроль, розподілене управління з прогнозуючими моделями (Model Predictive Control – MPC) із гарантіями уникнення зіткнень у роєвих польотах: оновлення траєкторій та обмін даними між апаратами виконуються лише за подіями (порушення зон безпеки/обмежень), що різко скорочує обчислення та мережевий трафік без погіршення безпеки. Робота [1] розв’язує задачу стеження з наперед визначеним часом (prescribed-time tracking) для множини апаратів за обмежень стану; подієві правила визначають моменти оновлення керувальних впливів, забезпечується збіжність у заздалегідь призначений час (prescribed-time convergence), а кількість оновлень істотно менша, ніж у періодично активованих підходах (time-triggered). Паралельно активно розвивається подієве сприйняття. Великий огляд з візуалізацією на основі подій [8] узагальнює властивості сенсорів змін яскравості (мікросекундні латентності, широкий діапазон освітленості, дуже низьке енергоспоживання) та демонструє їх синергію з реактивними контурами керування. Нещодавні системні роботи фокусуються на трактах обробки подій із малою затримкою: у [9] запропоновано масштабовану потокову обробку подієвого відео з мінімізацією затримок у каналі «сенсор → обчислення», а [10] показує приклад керування квадрокоптером у геометрично складному каналі (вузька труба) на основі подієвої велосиметрії – що підкреслює практичну важливість реактивної обробки для високодинамічних маневрів.

На рівні системного програмування класичні результати [11] пояснюють, коли активне опитування може переважати переривання за пропускною здатністю (наприклад, високошвидкісне введення-виведення без частих контекстних перемикань), а сучасні дослідження енергетики показують сценарії, де активне опитування має нижчі витрати енергії за умови виділеного ядра та великих черг NVMe (Non-Volatile Memory Express) [12]. Ці роботи важливі як контрприклади: вони вказують, що перевага реактивності не є «безумовною» і залежить від конкретної архітектури виконання – отже, потрібні предметні порівняльні експерименти для конкретних шаблонів очікування подій.

Як формальний апарат опису і реалізації алгоритмів керування БПЛА використовуються керуючі E-мережі (Control E-Networks – CEN) [3], які вдало використовуються для моделювання систем інтернету речей та здатні однаково природно описувати як циклічні, так і подієві процеси керування, включно з синхронізацією потоків подій. Це робить CEN універсальним засобом реалізації схем керування різних типів.

**Виділення недосліджених частин загальної проблеми.** Попри значний прогрес у подієво-тригерному керуванні, бракує порівнянь схем реакції на події у вбудованих системах, а саме циклічного опитування (polling), або використання подієвих слухачів (reactive), що відтворюються експериментально під час реального виконання, а не лише у моделюванні чи на рівні законів керування. Існуючі огляди ETC та динамічних тригерів тільки узагаль-

нюють правила активації та гарантії стабільності [4-6], але не розглядають виконавчі шаблони (polling/interrupt/reactive) та їхні ресурсні наслідки у вбудованих керувачах загального призначення. Тож не з'ясованими лишаються такі питання, як:

1) *Відсутність прямих порівнянь під час виконання «циклічного» та «подієвого» підходів за тотожних сценаріїв.* Прикладні роботи для БПЛА зосереджуються на розподіленому МРС або спеціальних законах керування (уникнення зіткнень, координація, стеження з наперед визначеним часом) і оцінюють частоту оновлень та якість траєкторій, але не розкладають цей ефект на алгоритмічну ціну очікування події у коді контролера (кількість перевірок, середній/піковий рівень завантаження процесора, блокувальне/неблокувальне виконання) [7; 1]. Це залишає відкритим питання: що саме дає виграш – закон керування чи спосіб очікування події під час виконання.

2) *Стабільність та передбачуваність часу реакції у серіях повторів.* Огляди зосереджені на теоретичних межах, асимптотиках та збіжності, тоді як варіативність (міжквартильний розмах, показники найгіршого випадку) для різних схем очікування в публікаціях майже не вимірюється експериментально на вбудованих платформах [4; 6]. Для задач безпеки (уникнення зіткнень, аварійні режими) це критично.

3) *Експериментальна перевірка поведінки очікування подій.* У більшості робіт увага зосереджена на правилах активації (коли «оновлювати»), а прямі вимірювання реальної поведінки механізмів очікування трапляються нечасто [4; 5].

4) *Профілі використання ресурсів та енергоспоживання за різних тривалостей очікування.* Роботи з подієвого сприйняття показують вигоди сенсорного тракту (мікросекундні затримки, низьке споживання) [8-10], однак досі бракує комплексного вимірювання роботи керувача для коректного порівняння «циклічного» та «подієвого» підходів у тих самих завданнях. Системні праці про компроміс між опитуванням і перериваннями демонструють контрприклади для високошвидкісного введення-виведення [11; 12], але ці результати не перенесено на керування польотом і роєву координацію.

5) *Зв'язок формальних моделей та виконання.* Попри наявність формальної бази CEN для опису як циклічних, так і подієвих процесів із J/Y-синхронізацією [3], у наявних роботах майже відсутні зв'язки між формальними мережевими моделями та емпіричними метриками виконання (затримка реакції, стабільність, ресурсні витрати) в експериментах із відтворюваними результатами.

У підсумку, потребує заповнення прогалина, де та сама функціональна логіка керування тестується у двох схемах очікування (циклічна і подієва) – прямими вимірами стабільності реакції, масштабуванням щодо часу очікування та (за можливості) кількості агентів, емпіричною перевіркою асимптотичної складності, повним ресурсним профілем і узгодженням із формальною моделлю CEN. Саме цю прогалину заповнює ця робота, пропонуючи відтворювану постановку експерименту та показники, що безпосередньо порівнюють архітектури очікування подій.

**Метою статті є** експериментально й кількісно довести переваги подієво-орієнтованого підходу на основі слухачів порівняно з циклічним опитуванням у вбудованих системах керування роєм безпілотних літальних апаратів у формалізмі CEN.

**У процесі дослідження вирішували такі завдання:**

1. Спроєктувати два функціонально еквівалентні сценарії (циклічний та подієвий) у CEN та реалізувати відтворювані тести.
2. Провести базове порівняльне вимірювання за фіксованого часу очікування та зафіксувати ключові метрики (час реакції, процесорне навантаження, пам'ять, кількість операцій).
3. Виконати багатоточковий аналіз у діапазоні часових затримок (0,1-15 с) для виявлення трендів і масштабування.

4. Оцінити стабільність часу реакції (розкид, міжквартильний розмах (IQR), найгірший випадок) у серіях повторних запусків.

5. Перевірити алгоритмічну складність очікування події для циклічного та подієвого підходів.

6. Узагальнити результати у вигляді таблиць і графіків та сформулювати рекомендації до вибору схеми для ройових застосувань.

**Виклад основного матеріалу.**

**Опис сценарію керування засобами CEN.**

Керування роєм БПЛА – це фактично інтелектуальна обробка потоку подій, що містить аналіз повідомлень від інших агентів, фіксація досягнення контрольних позицій, реагування на екстрені сигнали сенсорів, врахування обмежень енергоживлення (наприклад, рівень заряду батареї). Реакція на такі події має бути майже миттєвою, інакше виникає затримка, яка безпосередньо впливає на безпеку та виконання місії. Своєю чергою, керуючі Е-мережі є різновидом мереж Петрі і дозволяють моделювати як послідовні, так і паралельні взаємодіючі процеси з урахуванням умов їх виконання. Тож їх використання як формального апарату для реалізації алгоритмів керування роєм БПЛА є очікуваним і практично доцільним.

Щоб виокремити вплив схеми опрацювання подій, а не законів керування чи інших компонентів, розглянемо типовий мікросценарій: якщо з'являються певні дані, то дозволяється перейти до наступної дії. Це відповідає, наприклад, ситуації, коли ціль розпізнана за даними відеокамери: поки події немає, активація наступних обчислень не має сенсу. Така локальна декомпозиція легко представляється в CEN: мережа дозволяє взяти невеликий фрагмент загальної моделі та дослідити його поведінку незалежно; масштабування решти системи (інші гілки, інші потоки) не змінює локальну семантику очікування.

При використанні CEN за традиційною схемою очікування події реалізується через цикл опитування глобальної змінної: процес регулярно перевіряє, чи настав потрібний стан, і лише тоді переходить далі. Це просто в реалізації, але створює зайві обчислення та додає певну затримку. На рис. 1 наведено фрагмент алгоритму керування, що реалізує циклічну схему реагування на подію (polling), представлений у вигляді дводольного графа з'єднаних позицій (P) та переходів (T, X, Y). Після ініціалізації (перехід T1) процес входить у цикл із мінімальною паузою (перехід T3), де перехід X1 перевіряє глобальну змінну: якщо умова істинна – потік іде до «успішної» гілки P4–T2–P5, інакше – повертається в черговий оберт.

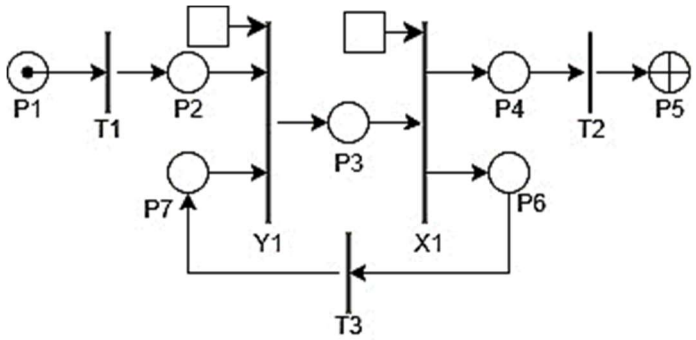


*Рис. 1. Схема циклічного CEN-фрагмента очікування події (polling)*

На рис. 2 показано реактивний фрагмент (reactive): після ініціалізації (T1) використовується перехід J1, що не блокує інші гілки та спрацьовує лише за наявності токенів на всіх своїх входах.

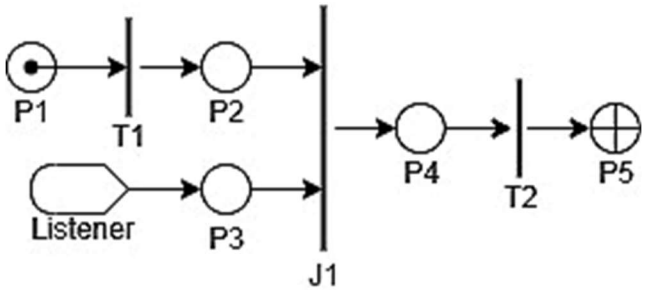


*Рис. 2. Схема реактивного CEN-фрагмента зі слухачем та переходом J (reactive)*

Токен від слухача подій заноситься в P3; як тільки він з'являється, виконується локальна перевірка умови спрацьовування J1. Важлива особливість нашої реалізації CEN: тут немає періодичного опитування; перевірка тригера J відбувається виключно в момент надходження токена до будь-якої вхідної позиції. Тому навіть для J із двома входами достатньо однієї перевірки: перший токен «позначає готовність» частини передумов, а під час появи другого відбувається єдина перевірка, після якої перехід миттєво спрацьовує. Це передає сутність подієво-орієнтованого підходу – відсутність циклів очікування.

**Середовище виконання та відтворюваність.**

Як апаратна платформа для реалізації алгоритму керування БПЛА у складі рою використовувався додатковий мікрокомп'ютер Jetson Nano, на якому можна відтворити паралельне виконання процесів реагування, планування та зв'язку, притаманних агентові у складі мультиагентної системи. Програмна реалізація алгоритму керування, представленого у вигляді CEN, здійснювалась мовою Python 3.8.

Для зіставлення результатів натурних експериментів обидва сценарії (циклічний та реактивний) запускалися в окремих процесах (*subprocess*), а вимірювання виконувалися засобами psutil та високоточними таймерами (*time.perf_counter*). Фіксувалися час реакції, кількість операцій очікування (кількість повторів циклу або спрацьовування J переходу), профіль навантаження ЦП, RSS-пам'ять. RSS-пам'ять (Resident Set Size) – обсяг фізичної оперативної пам'яті, що фактично зайнятий процесом. На відміну від віртуальної пам'яті (VMS), RSS показує реальне споживання системних ресурсів.

Між запусками коду реалізації CEN встановлювалися короткі стабілізаційні паузи для згладжування фонового шуму. Крім цього, CEN-реалізацію було доповнено збиранням додаткової статистики: фіксувався час виникнення події (появи токенів у вхідних позиціях), кількість спрацювань переходу X у циклі, кількість спрацьовувань переходу J, кількість згенерованих/спожитих токенів і службові події. Усі ці дані зберігалися у JSON-логах та агрегувалися скриптами для побудови графіків, що забезпечувало наочне представлення результатів.

**Експеримент 1. За фіксованого часу очікування.**

Після реалізації двох наведених вище рівнофункціональних фрагментів схем керування (див. рис. 1-2) перевіряли, наскільки схема очікування впливає на поведінку керувача. Під час запуску обох реалізацій Jetson Nano у відокремлених процесах вимірювали:

- час реакції (мс) – прямий показник безпеки та здатності системи реагувати в реальному часі;
- середнє навантаження процесора (%) – за профілем навантаження;
- кількість операцій очікування (повтори циклу або одиничне спрацьовування переходу J), що може розглядатися як наближена оцінка алгоритмічної складності;
- RSS-пам'ять (МБ) – щоб перевірити, що виграш досягається не ціною додаткової постійної пам'яті.

Результати базового порівняння циклічної та подієво-орієнтованої схем керування за фіксованого часу проведення експерименту наведено на рис. 3. Реактивна реалізація показує у 183 рази менший час реакції (8,301 мс проти 0,045 мс), у 235 разів нижчу середню завантаженість ЦП (22,97 % → 0,098 %), зменшення кількості операцій очікування у 21 раз (21 перевірка в циклі проти одного спрацювання переходу J), використання пам'яті – без суттєвих відмінностей (приблизна 20 та 21 МБ). Це підтверджує висунуту гіпотезу, що відмова від циклу прибирає зайві обчислення та пов'язану з ними затримку.

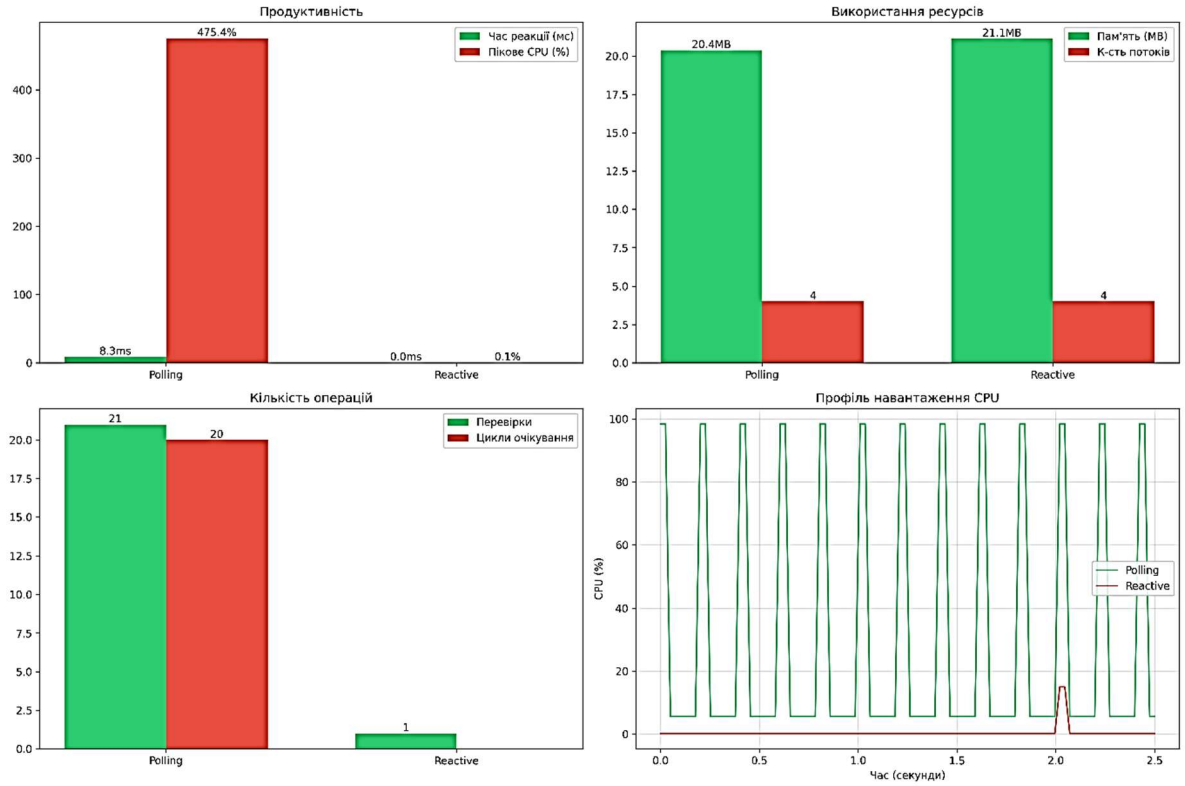


*Рис. 3. Результати порівняння за фіксованого часу очікування*

**Експеримент 2. За змінного часу очікування.**

Ідея експерименту полягає у дослідженні того, як змінюється поведінка обох схем керування зі зростанням інтервалу очікування. Це важливо для ройових місій різної тривалості та для подій із рідкісною появою. Проводили вимірювання у 20 точках діапазону часу очікування 0,1-15 с; кожна точка – окремий процес зі стабілізаційними паузами. Оцінювали:

- Час реакції – наскільки швидко система відповідає на подію. Критично для безпеки дронів.
- Навантаження процесора – скільки ресурсу процесора споживає система. Впливає на енергію батареї та час польоту.
- Кількість перевірок – скільки операцій виконує система. Циклічний підхід робить багато перевірок (циклічно кожні 0,1 с), а подієвий – лише одну на подію.
- RSS-пам'ять – скільки фізичної пам'яті використовує процес. Важливо для обмежених ресурсів дронів.

З рис. 4 видно, що крива подієвої реалізації (reactive) практично пласка, тоді як у циклічної (polling) вона зростає разом із часом очікування. Подієва реалізація стабільно швидка (≈ 0,00005 с), а циклічна повільніша та погіршується з часом (від 0,0004 с до 0,074 с). Середня перевага подієвого підходу: 756 разів. Середня навантаженість ЦП у подієвого підходу нижча приблизно у 157 разів. Графік залежності кількості перевірок

від часу очікування для циклічного підходу лінійний, що відповідає принципу: що довше очікування, то більше операцій O(n*t/Δt), де n = 1 (однорівневий цикл), t – заданий час очікування, Δt – крок циклу опитування (≈ 0,1 с), далі використовуємо скорочення O(t/Δt); для подієвого підходу кількість операцій – константа (= 1), то алгоритмічна складність схеми керування дорівнює O(1).

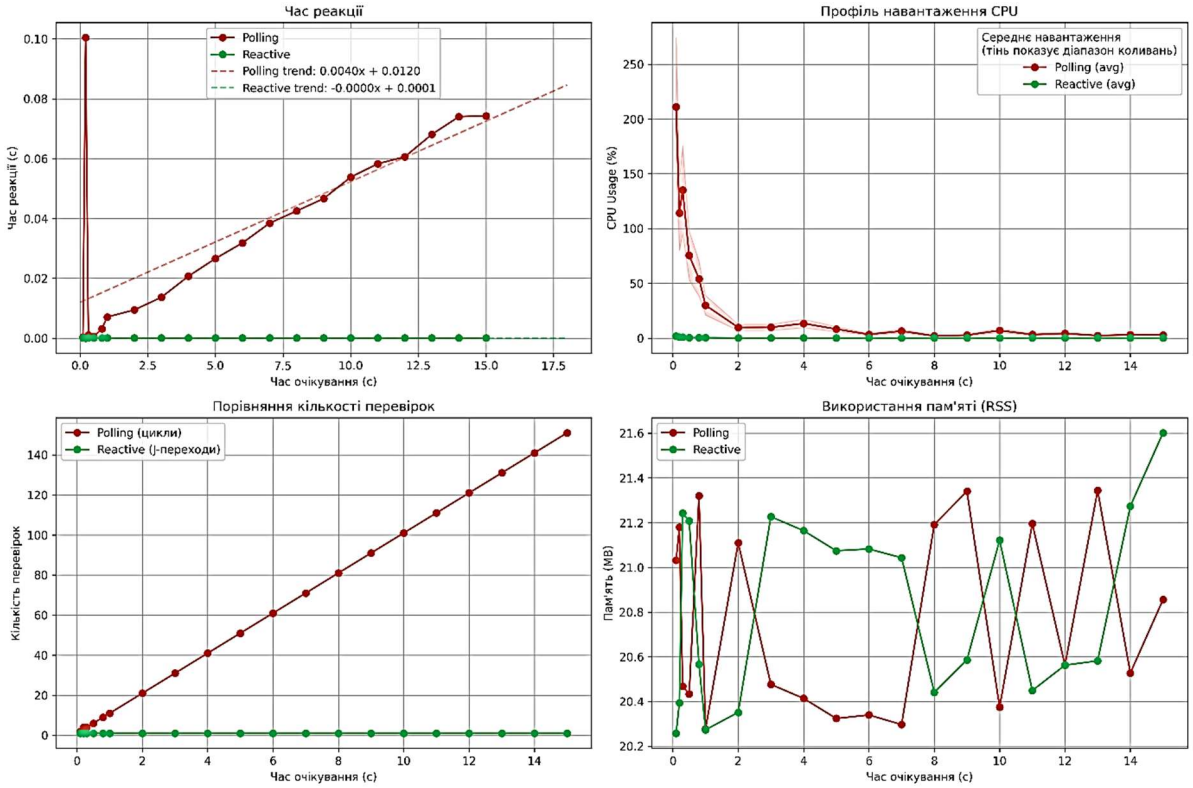


*Рис. 4. Результати порівняння за змінного часу очікування*

**Експеримент 3. Стабільність часу реакції.**

Надійність ройового керування визначається не лише середнім часом відгуку, а й стабільністю цього відгуку в умовах фонового навантаження. Нестабільність виникає через недетермінованість виконання на борту (планувальник ОС, переривання, конкуренція за процесор з процесами сприйняття та зв'язку, кеш-ефекти, енергозбереження, збірка сміття в Python). У циклічному підході постійні перевірки додатково завантажують ядра, підсилюючи варіативність; подієва обробка, навпаки, усуває холості обчислення.

Щоб кількісно оцінити стабільність, ми 30 разів повторили вимірювання часу реакції для кожної схеми на одному й тому самому таймінгу (~ 2 с). Спершу розглядали зведені графіки розподілу значень – вони показують, де зосереджена більшість вимірів і чи трапляються поодинокі дуже великі або дуже малі значення. Далі використовували такі показники:

- Міжквартильний розмах (IQR). Це ширина «середньої половини» вимірів: від значення, нижче якого лежить 25 % усіх вимірів (25-й процентиль), до значення, нижче якого лежить 75 % вимірів (75-й процентиль). Чим менший IQR, тим менша типова мінливість між запусками, тобто система стабільніша.
- Діапазон (макс − мін) – груба оцінка розкиду в найгіршому випадку, чутлива до поодиноких довгих затримок;
- Тест Левена. Статистична перевірка того, чи однаковий розкид значень у двох групах (циклічний та подієвий підходи). Мале значення p означає, що розкиди дійсно різні.

- Перекіс у бік великих затримок. Додатково якісно оцінювали, чи не з'являються рідкісні, але дуже великі затримки частіше в однієї зі схем; це видно на графіках як поява поодиноких великих значень праворуч шкали.

Як показано на рис. 5, реактивний підхід має у 19 разів вужчий діапазон і у 11 разів менший IQR, ніж циклічний; тест Левена дає $p \approx 1{,}6 \cdot 10^{-8}$, тобто різниця дисперсій двох розподілів часу реакції статистично значуща. Інакше кажучи, подієві J-переходи не лише пришвидшують реакцію, а й зменшують випадкові коливання часу затримки, що критично для усунення ситуацій найгіршого випадку. Практично це означає суттєво менший ризик поодиноких великих затримок, що безпосередньо впливає на надійність уникнення зіткнень і відпрацювання екстрених команд у рої.

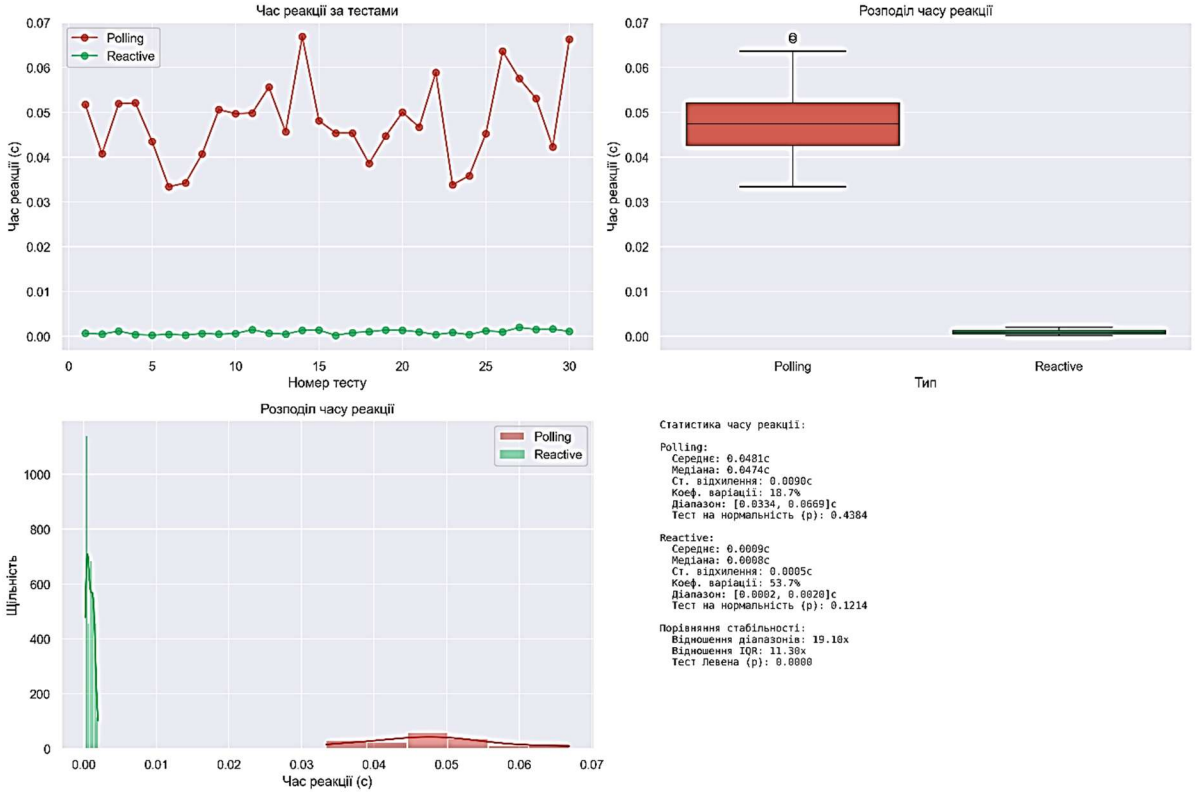


*Рис. 5. Порівняння стабільності часу реакції для циклічного та реактивного підходів*

**Експеримент 4. Експериментальна перевірка алгоритмічної складності очікування.**

Мета цього експерименту – з'ясувати, скільки саме операцій очікування виконує кожен підхід залежно від тривалості очікування події. Відповідно до результатів експерименту 1, для циклічного підходу очікуємо $O(t/\Delta t)$ (цикл встигає зробити $t/\Delta t$ перевірок), для подієвого – $O(1)$ (подія ініціює єдине спрацювання J). Вимірювали кількість перевірок (для циклу) або одиничні спрацювання J (для подієвого) як безпосередню оцінку алгоритмічної складності; додатково аналізували залежність часу реакції від часу очікування на предмет залежності від t.

Результати (рис. 6) відповідають очікуванню. Для циклічного підходу отримано майже ідеальну лінійну залежність між числом перевірок і часом очікування: $R^2 \approx 0{,}9999$, нахил $\approx 9{,}7^{-10}$ перевірок/с, що узгоджується з $\Delta t \approx 0{,}1$ с і означає $O(t/\Delta t)$. Для подієвого підходу кількість операцій у всіх точках дорівнює 1, тобто $O(1)$; при цьому час реакції практично не залежить від t.

Цей експеримент дає емпіричне пояснення тому, що ми спостерігали раніше: у циклічному підході з ростом t зростає число перевірок, відповідно – навантаження на процесор і затримка; у реактивному підході вони залишаються сталими, тому виграш за часом реакції та за ресурсами зростає разом із тривалістю очікування.

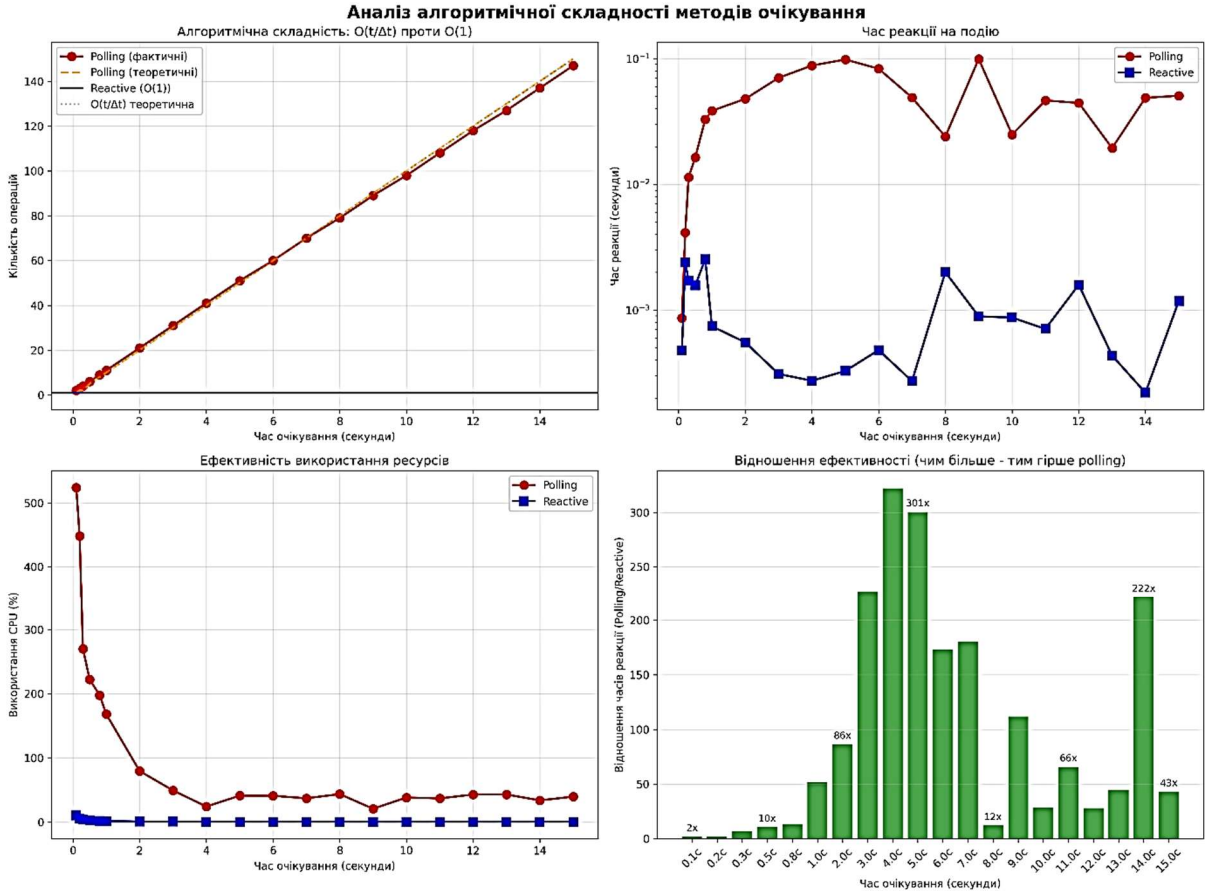


*Рис. 6. Результати перевірки поведінки очікування події та кількості операцій*

**Висновки.** У роботі представлено реалізацію подієво-орієнтованої схеми керування БПЛА у складі рою та її порівняння зі схемою циклічного керування.

Порівняння виконано на рівнофункціональних фрагментах алгоритмів керування, формалізованих за допомогою CEN та реалізованих мовою Python у відокремлених процесах на апаратній платформі Jetson Nano.

Проведені натурні експерименти дали змогу оцінити внесок схем керування у швидкодію, стабільність і ресурсні витрати без домішок інших чинників.

Отримані результати демонструють, що відмова від циклів опитування на користь слухачів та J-переходів CEN істотно покращує часові характеристики та профіль навантаження. За фіксованого часу подієва реалізація забезпечує приблизно у 183 рази менший час реакції і у 235 разів нижчий середній рівень навантаження процесора, при цьому кількість операцій очікування зменшується у 21 раз. У багатоточковому порівнянні на змінному діапазоні часу затримки 0,1-15 с зберігається та сама тенденція: затримка подієвої схеми практично не залежить від інтервалу очікування, тоді як затримка циклічної схеми деградує зі зростанням цього інтервалу; співвідношення часу реакції (циклічна/подієва реалізація) має медіанне значення близько 719, середнє – 756, максимальне – 2 539; середнє завантаження процесора для подієвого підходу нижче орієнтовно у 157 разів.

Не менш важливою є передбачуваність поведінки. Серії повторних запусків за однакових умов засвідчили помітне зменшення варіативності для подієвого підходу: діапазон значень часу реакції вужчий приблизно у 19 разів, міжквартильний розмах – у 11 разів; відмінність дисперсій статистично значуща ($p \approx 1{,}6 \cdot 10^{-8}$). Таким чином, реагування не лише пришвидшується, а й стає стабільнішим у типових і найгірших ситуаціях, що є важливим для критичних сценаріїв у рої.

Аналіз алгоритмічної складності пояснює спостережувані виграші: для циклів опитування кількість перевірок зростає пропорційно часу очікування з майже ідеальною лінійністю, тоді як для подієвої реалізації залишається сталою. Ця різниця безпосередньо відбивається на затримці та енергоспоживанні, особливо в умовах багатозадачності на борту.

Практичний висновок для вбудованих систем керування роями полягає в доцільності впровадження подієвої схеми керування як стандартної у всіх випадках, де домінує очікування зовнішніх або внутрішніх подій (надходження повідомлень від інших агентів, підтвердження досягнення позиції, екстрені команди, сигнали обмежень і станів енергоживлення). Це дозволяє одночасно зменшити затримку та коливання затримки, знизити середні та пікові витрати процесорного часу і, як наслідок, покращити енергоефективність та масштабованість під час виконання довготривалих місій. Використання циклічного опитування може бути виправданим у вузьких високошвидкісних підсистемах введення-виведення, де експериментально доведено кращу пропускну здатність або енергетичний профіль. Але й у таких випадках доцільні гібридні схеми з локальним циклічним опитуванням у поєднанні з подієвою координацією.

Сформульовані висновки спираються на узгоджені результати чотирьох незалежних серій експериментів і відтворювану методику вимірювань, що підвищує їхню достовірність. Водночас межі зовнішньої валідності визначаються вибраною платформою (Jetson Nano) та програмним середовищем Python 3.8; перенесення на інші апаратні платформи є предметом подальших досліджень. Перспективні напрями подальших досліджень: інтеграція подієвої архітектури з розподіленими протоколами взаємодії в роєвих системах та енергометрія на рівні апаратної платформи для прогнозування тривалості виконання місії.

## Список використаних джерел

1. Han, X., Yu, P., Lv, M., Shi, Y., & Wang, N. (2025). Event-driven prescribed-time tracking control for multiple UAVs with flight state constraints. *Machines, 13*(3), 192. https://doi.org/ 10.3390/machines13030192.

2. Vitale, A., Renner, A., Nauer, C., Scaramuzza, D., & Sandamirskaya, Y. (2021, May). Event-driven vision and control for UAVs on a neuromorphic chip. In *2021 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 103-109). IEEE. https://arxiv.org/abs/2108.03694.

3. Kazymyr, V., Pasko, O., Usik, A., & Sydorenko, D. (2019). New paradigm of model-oriented control in IoT. In R. Damaševičius & G. Vasiljevienė (Eds.), *Information and software technologies: ICIST 2019* (*Communications in Computer and Information Science*, Vol. 1078). Springer, Cham. https://doi.org/10.1007/978-3-030-30275-7_46.

4. Zhang, X.-M., Han, Q.-L., Ge, X.-H., *et al.* (2025). An overview of recent advances in event-triggered control. *Science China Information Sciences, 68*(6), 161201. https://doi.org/10.1007/s11432-024-4437-9.

5. Ge, X., Han, Q.-L., & Dong, H. (2021). Dynamic event-triggered control and estimation: A survey. *International Journal of Automation and Computing, 18*, 1294-1316. https://doi.org/10.1007/s11633-021-1306-z.

6. Liu, W., Zhang, B., & Jiang, H. (2024). A review of event-triggered consensus control in multi-agent systems. *Engineering Applications of Artificial Intelligence.* https://doi.org/10.1080/23307706.2024.2388551.

7. Gräfe, A., Eickhoff, J., & Trimpe, S. (2022). *Event-triggered and distributed model predictive control for guaranteed collision avoidance in UAV swarms*. *arXiv preprint* arXiv:2206.11020. https://doi.org/10.48550/arXiv.2206.11020.

8. Gallego, G., Delbrück, T., Orchard, G., Bartolozzi, C., Taba, B., Censi, A., … Liu, S.-C. (2022). Event-based vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 44*(1), 154-180. https://doi.org/10.1109/TPAMI.2020.3008413.

9. Hamara, A., Kilpatrick, B., Baratta, A., Kofink, B., & Freeman, A. C. (2024). Low-latency scalable streaming for event-based vision. *arXiv preprint arXiv:2412.07889*. https://arxiv.org/abs/2412.07889.

10. Bauersfeld, L., & Scaramuzza, D. (2025). Low-latency event-based velocimetry for quadrotor control in a narrow pipe. *arXiv preprint*, arXiv:2507.15444. https://arxiv.org/abs/2507.15444.

11. Yang, J., Minturn, D. B., & Hady, F. (2012, February). When poll is better than interrupt. In *Proceedings of the 10th USENIX Conference on File and Storage Technologies (FAST '12)*. San Jose, CA, USA: USENIX Association. Retrieved from https://www.usenix.org/conference/fast12/when-poll-better-interrupt.

12. Harris, B., & Altiparmak, N. (2022, July). When poll is more energy efficient than interrupt. In *Proceedings of the 14th ACM Workshop on Hot Topics in Storage and File Systems (HotStorage '22)* (pp. 59-64). New York, NY: Association for Computing Machinery. https://doi.org/10.1145/3538643.3539747.

## References

1. Han, X., Yu, P., Lv, M., Shi, Y., & Wang, N. (2025). Event-driven prescribed-time tracking control for multiple UAVs with flight state constraints. *Machines, 13*(3), 192. https://doi.org/ 10.3390/machines13030192.

2. Vitale, A., Renner, A., Nauer, C., Scaramuzza, D., & Sandamirskaya, Y. (2021, May). Event-driven vision and control for UAVs on a neuromorphic chip. In *2021 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 103-109). IEEE. https://arxiv.org/abs/2108.03694.

3. Kazymyr, V., Pasko, O., Usik, A., & Sydorenko, D. (2019). New paradigm of model-oriented control in IoT. In R. Damaševičius & G. Vasiljevienė (Eds.), *Information and software technologies: ICIST 2019* (*Communications in Computer and Information Science*, Vol. 1078). Springer, Cham. https://doi.org/10.1007/978-3-030-30275-7_46.

4. Zhang, X.-M., Han, Q.-L., Ge, X.-H., *et al.* (2025). An overview of recent advances in event-triggered control. *Science China Information Sciences, 68*(6), 161201. https://doi.org/10.1007/s11432-024-4437-9.

5. Ge, X., Han, Q.-L., & Dong, H. (2021). Dynamic event-triggered control and estimation: A survey. *International Journal of Automation and Computing, 18*, 1294-1316. https://doi.org/10.1007/s11633-021-1306-z.

6. Liu, W., Zhang, B., & Jiang, H. (2024). A review of event-triggered consensus control in multi-agent systems. *Engineering Applications of Artificial Intelligence.* https://doi.org/10.1080/23307706.2024.2388551.

7. Gräfe, A., Eickhoff, J., & Trimpe, S. (2022). *Event-triggered and distributed model predictive control for guaranteed collision avoidance in UAV swarms*. *arXiv preprint* arXiv:2206.11020. https://doi.org/10.48550/arXiv.2206.11020.

8. Gallego, G., Delbrück, T., Orchard, G., Bartolozzi, C., Taba, B., Censi, A., … Liu, S.-C. (2022). Event-based vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 44*(1), 154-180. https://doi.org/10.1109/TPAMI.2020.3008413.

9. Hamara, A., Kilpatrick, B., Baratta, A., Kofink, B., & Freeman, A. C. (2024). Low-latency scalable streaming for event-based vision. *arXiv preprint arXiv:2412.07889*. https://arxiv.org/abs/2412.07889.

10. Bauersfeld, L., & Scaramuzza, D. (2025). Low-latency event-based velocimetry for quadrotor control in a narrow pipe. *arXiv preprint*, arXiv:2507.15444. https://arxiv.org/abs/2507.15444.

11. Yang, J., Minturn, D. B., & Hady, F. (2012, February). When poll is better than interrupt. In *Proceedings of the 10th USENIX Conference on File and Storage Technologies (FAST '12)*. San Jose, CA, USA: USENIX Association. Retrieved from https://www.usenix.org/conference/fast12/when-poll-better-interrupt.

12. Harris, B., & Altiparmak, N. (2022, July). When poll is more energy efficient than interrupt. In *Proceedings of the 14th ACM Workshop on Hot Topics in Storage and File Systems (HotStorage '22)* (pp. 59-64). New York, NY: Association for Computing Machinery. https://doi.org/10.1145/3538643.3539747.

Отримано 19.09.2025

*UDC 004.94:681.5*

***Petro Oleksiienko[1], Volodymyr Kazymyr[2]***

[1]PhD Student, Recipient of the Doctor of Philosophy degree in specialty 122
Institute of Mathematical Machines and Systems Problems of the Ukraine National Academy of Science (Kyiv, Ukraine)
**E-mail:** alexeenkop77@gmail.com. **ORCID:** https://orcid.org/0009-0009-9588-9490

[2]Doctor of Sciences, Professor, Professor of the Department of Information and Computer Systems
Chernihiv Polytechnic National University (Chernihiv, Ukraine)
**E-mail:** vvkazymyr@gmail.com. **ORCID:** https://orcid.org/0000-0001-8163-1119. **ResearcherID:** Q-2925-2016

# EVENT-ORIENTED APPROACH IN THE IMPLEMENTATION OF EMBEDDED UAV SWARM CONTROL SYSTEMS

*This paper presents an experimental, reproducible study of an event-driven (listener-based) approach for embedded control in drone swarms, contrasted with the classical cyclic polling used to monitor system states. Cyclic checks increase processor load, waste energy, and degrade scalability as agent count or mission time grows. To isolate the effect of the waiting mechanism itself (rather than a control law), both realizations are modeled in the Control E-Networks (CEN) formalism as functionally equivalent fragments with identical "wait for event → continue" logic.*

*Implementations ran on a Jetson Nano using Python 3.8. Each scenario executed in a separate process (subprocess isolation). The CEN runtime was instrumented with counters and time stamps (token arrivals, checks, J-firings); measurements used high-resolution timers and psutil. We recorded reaction time, number of waiting operations (polling checks vs. a single J-transition), average/peak CPU load, and RSS memory, storing raw JSON logs for verification.*

*We conducted four complementary experiments. First, a fixed-timing baseline contrasted latency, CPU load, waiting operations, and memory under identical conditions. Second, a multi-point study (0,1-15 s) examined how latency and CPU scale with longer waiting intervals and how the count of polling checks grows, while the event-driven variant remains event-triggered. Third, a repeatability study with multiple runs assessed predictability via the spread of reaction times and variance comparison. Fourth, an empirical examination of waiting behavior related the number of operations to the waiting interval and observed the corresponding trend in reaction time. Across all experiments, the event-driven realization consistently delivered markedly lower latency and CPU usage, fewer waiting operations, and tighter variability, whereas polling degraded as the waiting interval increased.*

*These findings show that replacing cyclic waiting with listener-based, event-driven execution in CEN improves responsiveness, resource efficiency, and predictability–properties essential for collision avoidance, emergency handling, waypoint confirmation, and long-duration missions under resource contention. Practically, event-driven mechanisms should be preferred in branches dominated by waiting for external or internal events, while cyclic polling may remain justified only in narrow high-throughput I/O subsystems where its advantage is demonstrated.*

***Keywords:*** *UAV swarms; multi-agent systems; Control E-Networks; control systems; event-driven control; listeners; cyclic polling.*

*Fig.: 6. References: 12.*

Олексієнко П. Д., Казимир В. В. Подієво-орієнтований підхід у реалізації вбудованих систем керування роєм бпла. *Технічні науки та технології,* 2025. № 3(41). С. 225-236. DOI: https://doi.org/10.25140/2411-5363-2025-3(41)-225-236.