Рішення разової спеціалізованої вченої ради
про присудження ступеня доктора філософії

Разова спеціалізована вчена рада Національного технічного університету «Дніпровська політехніка» (м. Дніпро) прийняла рішення про присудження ступеня доктора філософії **Стародубському Ігорю Петровичу** з галузі знань 12 «Інформаційні технології» за спеціальністю 122 «Комп'ютерні науки» на підставі публічного захисту дисертації на тему **«Методи та засоби переносимості програм та програмних систем на різні обчислювальні платформи»** «15» липня 2026 року.

Здобувач ступеня доктора філософії Стародубський Ігор Петрович народився 08 серпня 1986 р. у м. Кривий Ріг Дніпропетровської області.

У 2008 році закінчив Дніпровський національний університет за спеціальністю «Комп'ютерні науки», отримав кваліфікацію магістра з комп'ютерних наук. Наразі є аспірантом Національного технічного університету «Дніпровська політехніка» за спеціальністю 122 «Комп'ютерні науки» та навчається в аспірантурі у 2022–2026 роках.

За час навчання в аспірантурі Стародубський Ігор Петрович проявив себе відповідальним науковцем, здатним самостійно досліджувати предметну область, висувати наукові гіпотези, пропонувати нові методи розв'язання поставлених задач та підтверджувати їх ефективність шляхом проведення експериментів.

Стародубський Ігор Петрович має понад 20 років професійного досвіду у сфері інформаційних технологій.

Кафедра програмного забезпечення комп'ютерних систем високо оцінює особистісні та комунікативні якості Стародубського І. П., а також відзначає важливість і наукову новизну проведеного ним дисертаційного дослідження.

Індивідуальний план наукової роботи та тему дисертації здобувача Стародубського Ігоря Петровича було затверджено рішенням Вченої ради Національного технічного університету «Дніпровська політехніка» (протокол № 8 від 30 червня 2022 р.). Роботу виконано на кафедрі програмного забезпечення комп'ютерних систем. 22 травня 2026 року відбулося розширене засідання кафедри програмного забезпечення комп'ютерних систем (протокол № 12), на якому дисертацію рекомендовано до захисту в разовій спеціалізованій вченій раді. Рішенням Вченої ради Національного технічного університету «Дніпровська політехніка» (протокол № 16 від 28 травня 2026 р.) утворено разову спеціалізовану вчену раду з правом прийняття до розгляду та проведення захисту дисертації Стародубського Ігоря Петровича на здобуття ступеня доктора філософії з галузі знань 12 «Інформаційні технології» за спеціальністю 122 «Комп'ютерні науки».

Науковий керівник: доктор технічних наук, доцент, професор кафедри програмного забезпечення комп'ютерних систем Бердник Михайло Геннадійович.

Матеріали дисертаційної роботи повною мірою викладено у 17 наукових працях. Шість статей опубліковано у наукових виданнях, включених до переліку фахових видань України, усі індексуються у НМБД Index Copernicus, одна з них

– категорії А та індексується у НМБД Web of Science. Одинадцять наукових праць опубліковано у збірниках наукових праць та матеріалах міжнародних конференцій.

Результати дисертаційного дослідження впроваджено у ТОВ НВП «Агропромавтоматизація», іноземному підприємстві «SoftRequest LTD», Державному науково-виробничому підприємстві «Ельдорадо», ТОВ «Біологічні активні добавки», а також у навчальному процесі Національного технічного університету «Дніпровська політехніка». Усі впровадження підтверджені відповідними актами.

Дисертацію подано у вигляді спеціально підготовленого рукопису, українською мовою.

Наукова новизна одержаних результатів. Усі наукові положення дисертаційної роботи розроблено здобувачем самостійно.

Уперше:

- розроблено метод адаптивної контейнеризації з інтеграцією штучного інтелекту, який забезпечує автоматизоване та динамічне налаштування параметрів середовищ виконання програм відповідно до характеристик апаратно-програмної платформи, поточного стану системи та змінних умов навантаження. Метод, на відміну від існуючих підходів, поєднує контейнерні технології з алгоритмами машинного навчання, що дозволяє здійснювати інтелектуальну адаптацію параметрів контейнерів у реальному часі без необхідності ручного втручання;
- розроблено архітектуру інтелектуального управлінн контейнеризованими програмами у гетерогенних обчислювальних середовищах, яка передбачає взаємодію модулів аналізу середовища, систем моніторингу, алгоритмів штучного інтелекту та засобів оркестрації контейнерів. Архітектура забезпечує узгоджену роботу компонентів системи та дозволяє автоматично оптимізувати використання обчислювальних ресурсів у багатоплатформних середовищах;
- розроблено інформаційну технологію застосування машинного навчання для прогнозування потреб програм у ресурсах обчислювальної платформи на основі аналізу історичних даних та поточних метрик продуктивності, що дозволяє реалізувати проактивне управління контейнеризованими середовищами виконання та зменшити ймовірність деградації продуктивності або перевантаження системи.

Удосконалено:

- інформаційну технологію забезпечення переносимості програмних систем шляхом поєднання контейнерних технологій із методами машинного навчання, що дозволило перейти від статичних конфігурацій середовищ виконання до адаптивних, здатних враховувати специфіку апаратної архітектури, операційної системи та умов експлуатації програмного забезпечення;

– інформаційну технологію автоматизації процесів розгортання, масштабування та супроводу програмних систем у мультиплатформних і мультихмарних середовищах за рахунок інтеграції інтелектуальних механізмів прийняття рішень у процеси оркестрації контейнерів. Це забезпечує підвищення ефективності використання ресурсів та зниження витрат на експлуатацію програмного забезпечення;
– метод управління контейнеризованими програмами у середовищах з обмеженими ресурсами, зокрема у вбудованих та периферійних обчислювальних системах, шляхом використання адаптивних алгоритмів оптимізації, що дозволяють забезпечити стабільну роботу програм за умов обмеженої обчислювальної потужності.

Набули подальшого розвитку:

– методи забезпечення переносимості програм та програмних систем у гетерогенних обчислювальних середовищах за рахунок використання інтелектуальних механізмів аналізу та адаптації середовищ виконання, що розширює можливості застосування контейнеризації у складних багатоплатформних системах;
– інформаційні технології щодо використання штучного інтелекту для оптимізації продуктивності, стабільності та масштабованості програмних систем, які функціонують у динамічних середовищах з нерівномірним навантаженням та змінними вимогами до ресурсів;
– методи автоматизованого управління середовищами виконання програм у мультихмарних інфраструктурах, що сприяють зниженню залежності від конкретного постачальника хмарних сервісів та підвищенню гнучкості розгортання програмного забезпечення.

У дискусії взяли участь голова та всі члени разової спеціалізованої вченої ради:

Голова ради: **Алексєєв Михайло Олександрович**, доктор технічних наук, професор, завідувач кафедри програмного забезпечення комп'ютерних систем Національного технічного університету «Дніпровська політехніка».

Запитання № 1: У чому полягає основна наукова новизна запропонованого методу адаптивної контейнеризації порівняно з традиційним використанням контейнерів як статичних одиниць розгортання?

Запитання № 2: Яким чином у дисертації поєднано адаптацію на рівні компіляції та адаптацію на рівні виконання і який практичний ефект забезпечує їх спільне застосування?

Відповіді здобувача **Стародубського І. П.** на запитання:

Відповідь на запитання № 1: Основна новизна полягає в тому, що контейнеризована програмна система у роботі розглядається як керований динамічний об'єкт. Контролер безперервно збирає телеметрію, формує поточний стан системи, оцінює відхилення переносимості D(t), прогнозує наслідки

допустимих керуючих дій і через Kubernetes control plane змінює ресурсні параметри, кількість реплік або політики розміщення. Традиційна контейнеризація забезпечує ізоляцію та відтворюваність середовища, але сама по собі не виконує такої інтелектуальної адаптації.

Відповідь на запитання № 2: На етапі компіляції генетичний алгоритм підбирає конфігурацію параметрів трансляції для цільової платформи, що дає змогу зменшити час виконання, енергоспоживання та використання пам'яті. На етапі виконання адаптивний контролер реагує на фактичне навантаження і стан середовища. Отже, build-time оптимізація зменшує базове ресурсне навантаження, а run-time адаптація підтримує стабільність системи за динамічних змін.

Рецензенти:

Приходченко Сергій Дмитрович, доцент кафедри програмного забезпечення комп'ютерних систем Національного технічного університету «Дніпровська політехніка», кандидат технічних наук, доцент. Рецензія є позитивною. Зауваження та дискусійні положення:

1. У роботі недостатньо формалізовано математичний апарат, який лежить в основі функції оцінювання відхилення переносимості D(t): не повною мірою розкрито принципи нормалізації різнорідних параметрів та методику вибору вагових коефіцієнтів;

2. Недостатньо уваги приділено теоретичному аналізу стійкості адаптивного контуру керування — у роботі відсутні формальні докази збіжності або стабільності системи в умовах динамічної зміни параметрів середовища функціонування;

3. Потребує уточнення роль методів машинного навчання у запропонованій архітектурі: з тексту роботи не завжди однозначно зрозуміло, чи використовуються моделі машинного навчання як допоміжний евристичний механізм, чи як повноцінний елемент системи прийняття рішень;

4. Експериментальну частину роботи доцільно було б розширити шляхом порівняльного аналізу з існуючими аналогами;

5. Запропонований метод, орієнтований на мінімізацію відхилення переносимості D(t), за певних сценаріїв може супроводжуватися підвищеним споживанням обчислювальних ресурсів, що в умовах хмарного розгортання здатне збільшувати вартість експлуатації; доцільно було б розглянути врахування вартісного (економічного) критерію поряд із критерієм переносимості;

6. Ефективність методу залежить від якості та повноти історичних даних, на яких навчаються моделі машинного навчання, тому за появи нових, нетипових сценаріїв навантаження точність оцінювання та прогнозування D(t) може знижуватися.

Вважаю, що висловлені зауваження не є визначальними, не зменшують загальну наукову новизну та практичну значимість результатів і не впливають на позитивну оцінку дисертаційної роботи.

Запитання № 1: Яку саме функцію виконують моделі машинного навчання у запропонованому керуючому циклі?

Запитання № 2: Чому запропонований метод не потребує внесення змін до прикладного коду програмної системи?

Відповіді здобувача **Стародубського І. П.** на зауваження:

Щодо недостатньої формалізації функції D(t). У роботі вона визначена як зважена сума нормалізованих відхилень експлуатаційних метрик, а ваги відображають їхню пріоритетність. Водночас додатковий аналіз чутливості до ваг і порогових значень підвищив би відтворюваність результатів. Методи машинного навчання є повноцінним аналітичним компонентом: вони оцінюють і прогнозують D(t), а остаточне рішення приймається політикою керування з урахуванням обмежень оркестратора. Порівняння проведено з базовим режимом, однак розширення аналізу сучасними аналогами є доцільним.

З іншими зауваженнями погоджуюся. Формальний доказ стійкості не був окремою задачею роботи, хоча практична стабільність забезпечувалася обмеженням допустимих дій, механізмами hysteresis і cooldown та підтверджена експериментально. Метод може збільшувати споживання ресурсів, оскільки його основною метою є мінімізація D(t), тому надалі доцільно врахувати економічний критерій. Також якість прогнозування залежить від історичних даних, що обґрунтовує необхідність періодичного донавчання моделей і виявлення нетипових режимів. Зазначені зауваження визначають напрями подальшого розвитку.

Відповіді здобувача **Стародубського І. П.** на запитання:

Відповідь на запитання № 1: Моделі машинного навчання прогнозують стан програмної системи після застосування кожної допустимої керуючої дії. Вони оцінюють очікуваний результат зміни ресурсних параметрів, кількості реплік або інших налаштувань контейнерного середовища. Остаточний вибір дії виконує контролер на основі цих прогнозів; самі моделі машинного навчання безпосередньо не змінюють конфігурацію системи.

Відповідь на запитання № 2: Метод працює на рівні контейнерного середовища та системи оркестрації. Він змінює параметри виділення процесорних ресурсів і пам’яті, кількість реплік, правила масштабування та розміщення контейнерів. Усі ці дії виконуються через стандартні механізми Kubernetes, тому прикладна логіка програми та її вихідний код залишаються незмінними.

Коряшкіна Лариса Сергіївна, заступник завідувача кафедри системного аналізу та управління, професор кафедри системного аналізу та управління Національного технічного університету «Дніпровська політехніка», доктор технічних наук, доцент. Рецензія є позитивною. Зауваження та дискусійні положення:

1. Математична формалізація еволюції програмної системи у вигляді (2.1), (2.2) містить неточність, адже профіль цільової обчислювальної платформи P,

який є складовою частиною вектору стану системи, такий, що, як зазначено на стор. 56, «Компоненти вектору є статичними (platform-level metadata)». Відтак, вектор P потрібно вилучити з вектору стану системи. Якщо ж деякі компоненти вектора P можуть змінюватися з часом (зокрема, обсяг доступної оперативної пам'яті), то в правих частинах (2.2) – (2.10) та усіх інших формулах, P є зайвим.

2. Алгоритм прийняття рішень (стор. 78), який представлений «з метою уточнення алгоритмічної сутності методу керування адаптивною контейнеризацією», бажано було б викласти з нумерацією етапів останнього. Ускладнює сприйняття викладеного матеріалу, по-перше, відмінність перших етапів (в алгоритму методу (стор. 76) – це генерація та фільтрація простору кандидатів, на виході – допустимі керуючи дії; в уточненому варіанті – це «збір метрик виконання контейнеризованої програмної системи у межах ковзного часового вікна W, в результаті чого формується поточна вектор-функція метрик m(t)»); по-друге – різне число етапів.

3. Використано одні й ті самі позначення для різних параметрів, зокрема, w(t) – вектор-функція зовнішніх збурень, зумовлених змінами навантаження; wi – вагові коефіцієнти в формулі (2.6); W – ковзне часове вікно.

4. Наведені в роботі експериментальні дослідження достатньо підтверджують ефективність запропонованого підходу, однак варто було б додатково підсилити емпіричну базу шляхом розширення набору тестових сценаріїв і конфігурацій обчислювальних платформ.

5. Вживання англомовної термінології дещо знижує читабельність тексту; наявні некоректні висловлення, зокрема, «абсолютні» цілі числа (стор. 57); трапляються поодинокі неточності та елементи, що мають редакційний характер (назва пункту 2.5.3 – окремо від самого тексту, підпис рисунку 2.4 – на окремій сторінці та ін.).

Зазначені зауваження не впливають на загальну позитивну оцінку дисертаційної роботи, її наукову новизну та практичну значущість.

Запитання № 1: Яку роль у запропонованому методі виконує профіль цільової обчислювальної платформи?

Запитання № 2: Чим відрізняються загальний алгоритм методу адаптивної контейнеризації та алгоритм однієї ітерації керуючого циклу?

Відповіді здобувача **Стародубського І. П.** на зауваження:

Щодо включення профілю платформи P до вектора стану системи. У роботі він використовується як контекстний параметр цільової платформи, а не як керована динамічна змінна. Для більш строгого математичного запису P доцільно винести з вектора стану x(t) і залишити параметром функцій переходу, оцінювання та прогнозування. Це уточнення підвищує формальну точність, але не змінює сутності методу. Зауваження щодо структури алгоритму також є слушним: у роботі подано загальний та деталізований варіанти опису, які не суперечать один одному, але їх доцільно було б об'єднати в єдину нумеровану послідовність — від збору метрик і формування стану до генерації, фільтрації,

прогнозування, вибору та застосування керуючої дії.

Повністю погоджуюся із зауваженням щодо близьких позначень w(t), wi та W. Для усунення неоднозначності зовнішні збурення доцільно позначити як η(t), а вагові коефіцієнти — як αi. Експериментальна перевірка охоплювала дві контрастні платформи та два характерні сценарії навантаження, однак розширення набору архітектур, типів програм і сценаріїв відмов посилило б емпіричну базу. Погоджуюся також із редакційними зауваженнями: частину англомовних термінів варто було замінити українськими відповідниками.

Відповіді здобувача **Стародубського І. П.** на запитання:

Відповідь на запитання № 1: Профіль цільової платформи описує її відносно сталі характеристики, зокрема архітектуру та кількість ядер процесора, номінальний обсяг оперативної пам’яті, наявність апаратних прискорювачів, а також характеристики операційної системи й середовища виконання. Він використовується для визначення еталонної поведінки програмної системи, формування допустимих керуючих дій та врахування платформних обмежень під час прийняття рішення. Поточне завантаження процесора, фактичне використання пам’яті та інші змінні характеристики належать до експлуатаційних метрик, а не до сталого профілю платформи.

Відповідь на запитання № 2: Загальний алгоритм описує повну послідовність роботи методу: від початкового налаштування системи та формування множини допустимих керуючих дій до запуску безперервного процесу адаптації. Алгоритм однієї ітерації деталізує лише повторювану частину цього процесу: збір телеметрії, формування поточного стану, оцінювання можливих дій, вибір керуючого впливу та його застосування. Тому ці алгоритми мають різну кількість етапів і різний рівень деталізації.

Офіційні опоненти:

1. **Вовна Олександр Володимирович**, професор кафедри програмних систем і технологій Київського національного університету імені Тараса Шевченка, доктор технічних наук, професор. Відгук є позитивним. Зауваження та дискусійні положення:

1. У дисертації переносимість значною мірою оцінюється через відхилення експлуатаційних показників від еталонного профілю. Водночас такі показники, як затримка, використання процесора, пам’яті або пропускна здатність, традиційно характеризують також продуктивність, ефективність і стабільність програмної системи. Тому доцільним було б детальніше обґрунтувати межі між поняттями переносимості, адаптивності та ефективності, а також чіткіше показати, за яких умов зменшення D(t) слід інтерпретувати саме як підвищення переносимості.

2. Формулювання наукової новизни у вступі не повністю охоплює результати четвертого розділу. Розроблений метод адаптивної компіляції на основі генетичного алгоритму фактично є самостійним науковим результатом, однак у переліку положень наукової новизни він не отримав достатньо чіткого

відображення. Доцільно було б узгодити формулювання наукової новизни, задач дослідження та загальних висновків із фактичним змістом четвертого розділу.

3. У функції відхилення переносимості D(t) використовуються вагові коефіцієнти, еталонні значення, допустимі інтервали та порогові параметри ε, T і ρ. У роботі зазначено, що їх вибір залежить від вимог програмної системи та рівня сервісу, проте методику визначення цих параметрів висвітлено недостатньо докладно. Додатковий аналіз чутливості результатів до зміни вагових коефіцієнтів і порогів посилив би обґрунтованість запропонованої моделі.

4. Для оцінювання та прогнозування D(t) передбачено застосування регресійних моделей, зокрема Random Forest, Gradient Boosting, лінійної регресії та неглибоких нейронних мереж. Водночас у дисертації бракує розгорнутого опису обсягу і структури навчальної вибірки, процедури поділу даних, налаштування гіперпараметрів та кількісних показників точності моделі, наприклад MAE, RMSE або R^2. Наведення таких даних дало б змогу повніше оцінити якість інтелектуального модуля та його здатність до узагальнення.

5. Експериментальну перевірку адаптивної контейнеризації проведено для двох платформних профілів і двох основних сценаріїв навантаження. Для ширшого підтвердження узагальнюваності результатів доцільно було б розширити експериментальну базу, включивши інші апаратні архітектури, класи прикладних програм, змішані навантаження та сценарії відмов. Крім того, бажано чітко зазначити кількість повторних запусків для експериментів третього розділу та навести довірчі інтервали одержаних оцінок.

6. Як базовий режим використано статичні ресурсні обмеження та типові механізми autoscaling Kubernetes. Порівняння з сучасними інтелектуальними засобами керування ресурсами, зокрема прогнозними алгоритмами масштабування, Vertical Pod Autoscaler або підходами на основі навчання з підкріпленням, дало б змогу точніше визначити переваги запропонованого методу. Доцільним також є кількісне оцінювання накладних витрат самого контролера та економічного компромісу між стабілізацією переносимості й додатковим споживанням ресурсів.

Висловлені зауваження мають переважно рекомендаційний і дискусійний характер. Вони не заперечують достовірності основних результатів, не знижують наукової цінності дисертації та можуть розглядатися як напрями подальшого розвитку запропонованих методів.

Запитання № 1: У чому полягає відмінність між підвищенням переносимості та звичайним підвищенням продуктивності програмної системи?

Запитання № 2: Які саме параметри контейнерного середовища може змінювати запропонований контролер під час адаптації програмної системи?

Відповіді здобувача **Стародубського І. П.** на зауваження:

Щодо розмежування переносимості, адаптивності та ефективності зазначу, що в роботі продуктивність і ресурсна ефективність розглядаються як експлуатаційні прояви переносимості, але не ототожнюються з нею. Зменшення

D(t) свідчить про підвищення переносимості лише тоді, коли поведінка системи на цільовій платформі наближається до еталонного або допустимого профілю без зміни прикладного коду. Із зауваженням щодо четвертого розділу погоджуюся: цю частину можна було б чіткіше відображений у науковій новизні, завданнях та загальних висновках. Щодо ваг, еталонів і порогів погоджуюся частково: вони фіксувалися до початку експериментів і не підбиралися під окремі сценарії, однак аналіз чутливості посилив би обґрунтованість моделі.

З іншими зауваженнями погоджуюся.

Відповіді здобувача **Стародубського І. П.** на запитання:

Відповідь на запитання № 1: Підвищення продуктивності зазвичай означає поліпшення окремих показників роботи програми на одній платформі, наприклад зменшення часу виконання або обсягу використаної пам’яті. У дисертації переносимість розглядається як здатність програмної системи зберігати функціональну коректність і допустимі експлуатаційні характеристики під час переходу між різними платформами. Тому метою керування є не максимізація продуктивності на одній платформі, а зменшення відмінностей у поведінці системи в різних апаратно-програмних середовищах.

Відповідь на запитання № 2: Контролер може змінювати ліміти процесорних ресурсів і оперативної пам’яті контейнера, кількість реплік програмного сервісу, параметри автоматичного масштабування та правила розміщення контейнерів на доступних обчислювальних вузлах. Вибрана керуюча дія застосовується через Kubernetes control plane, тому адаптація виконується на рівні середовища виконання без внесення змін до прикладного коду.

2. **Шекета Василь Іванович**, професор кафедри інженерії програмного забезпечення Івано-Франківського національного технічного університету нафти і газу, доктор технічних наук, професор. Відгук є позитивним. Зауваження та дискусійні положення:

1. Вибір оптимального керуючого впливу u*(t) на кожному кроці реалізовано жадібним (greedy) алгоритмом у дискретному просторі допустимих дій із застосуванням механізмів гістерезису та cooldown. Для повноти викладу корисним було б додаткове зіставлення такої стратегії з альтернативними підходами до вибору керуючої дії на просторі U(x,P) на окремих тестових сценаріях.

2. Програмний прототип контролера є централізованим компонентом керуючого циклу; питання його високої доступності та відмовостійкості (поведінка системи у разі відмови контролера, реплікація, тестування) у роботі розглянуто оглядово. Детальніше опрацювання цих аспектів є доцільним при доведенні прототипу до промислового рівня.

3. Часові параметри керуючого циклу — період ітерації Δ та ширина вікна спостереження W — описано якісно як компроміс між швидкістю реакції та стійкістю. Було б доцільним навести значення, використані в експериментах.

4. У роботі наведено результати щодо зниження енергоспоживання на

платформі ARM, отримані за показаннями вбудованих апаратних сенсорів. Для повноти викладу доцільно було б детальніше описати у тексті процедуру вимірювання та кількість повторних запусків, за якими отримано усереднені оцінки.

5. У роботі бракує кількісної оцінки накладних витрат, які вносить сам механізм адаптації (робота контролера, збір телеметрії засобами Prometheus) у споживання ресурсів керованої системи, що дозволило б оцінити співвідношення «вартість–ефект» запропонованого підходу.

6. У тексті місцями спостерігається непослідовність у використанні україномовної та англомовної термінології (частина термінів подається паралельно в обох формах) та трапляються поодинокі неточні формулювання, що мають редакційний характер і потребують уточнення.

Зазначені зауваження не є принциповими, стосуються переважно деталізації та повноти викладу й не знижують наукової та практичної цінності виконаного дослідження.

Запитання № 1: Чому для вибору керуючого впливу використано послідовне оцінювання допустимих дій, а не складніший оптимізаційний алгоритм?

Запитання № 2: Яким чином у запропонованому методі запобігають частим і взаємно протилежним змінам конфігурації контейнерів?

Відповіді здобувача **Стародубського І. П.** на зауваження:

Жадібний алгоритм обрано через дискретний та обмежений простір допустимих дій, що забезпечує швидке прийняття рішення в межах керуючого циклу. Водночас погоджуюся, що порівняння з багатокроковим прогнозним керуванням або reinforcement learning було б корисним. У прототипі період ітерації Δ становив 30 секунд, ширина вікна W — 30 секунд, а cooldown period — 60 секунд. Контролер мав дослідницький централізований характер: у разі його відмови система продовжує працювати в останньому стані, але адаптація припиняється. Для промислового застосування необхідні реплікація, leader election, health-checks та автоматичне відновлення.

З іншими зауваженнями погоджуюся. Процедуру вимірювання енергоспоживання на ARM, кількість повторів і усереднення результатів доцільно було описати детальніше. Накладні витрати контролера та Prometheus окремо не оцінювалися, тому надалі слід виміряти їхнє споживання CPU, пам’яті й мережі та визначити співвідношення «вартість–ефект».

Відповіді здобувача **Стародубського І. П.** на запитання:

Відповідь на запитання № 1: Множина допустимих керуючих дій у запропонованому методі є дискретною та обмеженою. До неї входять конкретні зміни ресурсних параметрів контейнерів, кількості реплік і політик розміщення. Контролер прогнозує результат кожної допустимої дії та обирає дію з найкращим очікуваним результатом. За обмеженої кількості кандидатів таке повне оцінювання не потребує окремого складного оптимізаційного алгоритму,

спрощує реалізацію та забезпечує однозначний вибір керуючої дії.

Відповідь на запитання № 2: Для цього застосовуються механізми гістерезису та cooldown. Гістерезис вимагає, щоб відхилення перевищило встановлений поріг, тому незначні короткочасні коливання метрик не спричиняють зміни конфігурації. Механізм cooldown блокує повторне застосування керуючої дії протягом заданого інтервалу після попередньої зміни, щоб система встигла відреагувати на неї. Разом ці механізми запобігають частим і взаємно протилежним змінам кількості реплік та ресурсних параметрів.

Результати відкритого голосування:

«За» 5 членів ради,

«Проти» нема членів ради.

На підставі результатів відкритого голосування разова спеціалізована вчена рада присуджує **Стародубському Ігорю Петровичу** ступінь доктора філософії з галузі знань 12 «Інформаційні технології» за спеціальністю 122 «Комп'ютерні науки».

Голова разової спеціалізованої вченої ради _____ Михайло АЛЕКСЄЄВ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ «ДНІПРОВСЬКА ПОЛІТЕХНІКА» * і.к. 02070743 *