Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра комп'ютерних наук
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

магістр
(назва освітнього ступеня)

на тему: Моделювання динамічних мереж міської інфраструктури з використанням кіберфізичних систем

Виконав: студент VI курсу, групи СТм-61
спеціальності 126 Інформаційні системи і технології
(шифр і назва спеціальності)

_____ Коваль А.А.
(підпис) (прізвище та ініціали)

Керівник _____ Гром'як Р.С.
(підпис) (прізвище та ініціали)

Нормоконтроль _____ Дуда О.М.
(підпис) (прізвище та ініціали)

Завідувач кафедри _____ Боднарчук І.О.
(підпис) (прізвище та ініціали)

Рецензент _____ _____
(підпис) (прізвище та ініціали)

Тернопіль
2023

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра комп'ютерних наук
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Боднарчук І.О.
(підпис) (прізвище та ініціали)
«28» _грудня 2023 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Магістр
(назва освітнього ступеня)

за спеціальністю 126 «Інформаційні системи і технології»
(шифр і назва спеціальності)

Студенту Коваль Андрій Андрійович
(прізвище, ім'я, по батькові)

1. Тема роботи Моделювання динамічних мереж міської інфраструктури з використанням кіберфізичних систем

Керівник роботи Гром'як Роман Сильвестрович, к.ф.-м.н., доцент кафедри КН
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від « 24 » листопада 2023 року № 4/7-1097

2. Термін подання студентом завершеної роботи 28 грудня 2023р.

3. Вихідні дані до роботи Наукові публікації про моделювання динамічних мереж міської інфраструктури, кіберфізичні системи та IoT пристрої.

4. Зміст роботи (перелік питань, які потрібно розробити)

Вступ. 1 Аналіз предметної області. 1.1 Еволюція кіберфізичних систем. 1.2 Парадигма КФС. 1.3 Інтернет речей. 1.4 Висновок до першого розділу. 2 Методології для проєктування КФС. 2.1 Основні виклики при розробці та інтеграції КФС. 2.2 Функціональний та архітектурний дизайн КФС. 2.3 Методи та інструменти для моделювання та імітації КФС. 2.4 Управління життєвим циклом продукту для КФС. 2.5 Архітектура та поведінкова парадигма для кіберфізичних систем. 2.6 Висновок до другого розділу. 3 Динамічна компонентна модель для кіберфізичних систем. 3.1 Динамічна компонентна модель Kevoree. 3.2 Концепція Kevoree для мікроконтролерів. 3.3 Практичне дослідження. 3.4. μ Kevoree. 3.5 Мікропрограма для підтримки основних змін. 3.6 Безшовна динамічна адаптація мікроконтролерів. 3.7 Обчислювальний експеримент. 3.7.1 Час простою. 3.7.2 Нестабільне використання пам'яті 3.7.3 Постійне використання пам'яті. 3.7.4 Затримка перезавантаження при відновленні. 3.8 Висновок до третього розділу. 4 Охорона праці та безпека в надзвичайних ситуаціях. 4.1 Питання щодо охорони праці. 4.2 Питання щодо безпеки в надзвичайних ситуаціях 4.3 Висновок до четвертого розділу. Висновок

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)

1 Титульна сторінка. 2 Тема, Мета, Об'єкт, Предмет дослідження. 3 Завдання дослідження. 4 Актуальність дослідження. 5 Процес переходу від мехатроніки до КФС та Інтернету речей. 6 Системні рівні абстракції між технологіями. 7 Дисциплінарна інтеграція на ранніх стадіях проєктування КФС. 8 Основні елементи управління життєвим циклом додатків. 9 Поточна архітектура КФС. 10 Відмовостійка архітектура КФС. 11 Первинні результати експерименту. 12 Процентний розподіл часу простою флеш-пам'яті. 13 Експеримент з ємністю SDRAM. 14 Вплив постійного розміру пам'яті на час завантаження 15 Висновки. 16 Завершальний слайд.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці	Сенчишин В.С., доцент		
Безпека в надзвичайних ситуаціях	Клепчик В.М., ст. викладач		

7. Дата видачі завдання 24 листопада 2023 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення з завданням до кваліфікаційної роботи	25.11.2023	*Виконано*
2.	Підбір наукових джерел про моделювання динамічних	26.11.2023-28.11.2023	*Виконано*
	мереж міської інфраструктури з використанням		
	кіберфізичних систем		
3.	Опрацювання наукових публікацій та збір даних по	29.11.2023-1.12.2023	*Виконано*
	темі роботи		
4.	Виконання дослідження згідно мети кваліфікаційної	2.12.2023-4.12.2023	*Виконано*
	роботи		
5.	Оформлення розділу «Аналіз предметної області»	5.12.2023-7.12.2023	*Виконано*
6.	Оформлення розділу «Методології для проєктування	8.12.2023-10.12.2023	*Виконано*
	КФС»		
7.	Оформлення розділу «Динамічна компонентна модель	11.12.2023-13.12.2023	*Виконано*
	для кіберфізичних систем»		
8.	Виконання завдання до підрозділу «Охорона праці»	14.12.2023-15.12.2023	*Виконано*
9.	Виконання завдання до підрозділу «Безпека в	16.12.2023-17.12.2023	*Виконано*
	надзвичайних ситуаціях»		
10.	Оформлення кваліфікаційної роботи	18.12.2023-19.12.2023	*Виконано*
11.	Нормоконтроль	19.12.2023-20.12.2023	*Виконано*
12.	Перевірка на плагіат	21.12.2023	*Виконано*
13.	Попередній захист кваліфікаційної роботи	22.12.2023	*Виконано*
14.	Захист кваліфікаційної роботи	29.12.2023	

Студент _____ Коваль А.А.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Гром'як Р.С.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Моделювання динамічних мереж міської інфраструктури з використанням кіберфізичних систем // Кваліфікаційна робота освітнього рівня «Магістр» // Коваль Андрій Андрійович // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра комп'ютерних наук, група СНн-61 // Тернопіль, 2023 // С. 72, рис. – 12, табл. – 3, кресл. – 16, додат. – 1, бібліогр. – 79.

Ключові слова: моделювання, мережа, кіберфізичні системи, архітектура програмного забезпечення, класифікація систем, вбудоване програмне забезпечення, обчислення, мікроконтролер.

Кваліфікаційна робота присв'ячена моделюванню динамічних мереж міської інфраструктури з використанням кіберфізичних систем.

В першому розділі кваліфікаційної роботи розглянуто еволюцію кфберфізичних систем. Висвітлено Інтернет речей як елемент КФС. Проаналізовано основні парадигми кібефізичних систем.

В другому розділі кваліфікаційної роботи описано основні виклики при розробці та інтеграції КФС. Досліджено архітектуру та поведінкову парадигму для КФС. Подано порівняльний опис функціонального та архітектурного дизайну кіберфізичних систем.

В третьому розділі кваліфікаційної роботи Розроблено концепцію Kevoree. Запропоновано динамічну компонентну модель Kevoree. Спроєктовано мікропрограму для підтримки основних змін. Протестовано безшовну динамічну адаптацію мікроконтролерів.

В четвертому розділі кваліфікаційної роботи розглянуто забезпечення безпечної роботи з обладнанням.

Об'єкт дослідження: динамічні мережі міської інфраструктури. Предмет дослідження: моделі та технології моделювання, використання кіберфізичних систем у контексті управління та оптимізації міської інфраструктури.

ANNOTATION

Modeling of the city infrastructure dynamic networks using cyber-physical systems // The educational level "Master" qualification work // Koval Andrii Andriiovych // Ternopil Ivan Pulyuy National Technical University, Faculty of Computer Information Systems and Software Engineering, Department of Computer Science, SNnm-61 group // Ternopil, 2023 // P. 72, fig. - 12, tables - 3, posters - 16, annexes - 1, ref. - 79.

Key words: modelling, network, cyber physical systems, software architect, system classification, embedded software, computing, microcontroller.

The qualification work is devoted to modelling dynamic networks of urban infrastructure using cyber-physical systems.

The first chapter of the qualification work discusses the evolution of cyber-physical systems. The Internet of Things as an element of CPS is highlighted. The main paradigms of cyber-physical systems are analysed.

The second section of the qualification work describes the main challenges in the development and integration of CPS. The architecture and behavioural paradigm for CPS are investigated. A comparative description of the functional and architectural design of cyber-physical systems is given.

In the third section of the qualification work, the Kevoree concept is developed. The dynamic component model of Kevoree is proposed. Firmware to support major changes was designed. Seamless dynamic adaptation of microcontrollers is tested.

The fourth section of the qualification work deals with ensuring safe operation of the equipment.

Object of research: dynamic networks of urban infrastructure. Subject of research: models and modelling technologies, the use of cyber-physical systems in the context of management and optimisation of urban infrastructure.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

API (англ. application programming interface) – Прикладний програмний інтерфейс.

ШІ – Штучний інтелект.

КФС – Кіберфізична система

IoT (англ. Internet of Things) – Інтернет речей.

SDRAM (англ. Synchronous Dynamic Random Access Memory) – синхронна динамічна пам'ять з довільним доступом.

EEPROM (англ. Electrically Erasable Programmable Read-Only Memory) – електронно програмована постійно читабельна пам'ять з можливістю стирання.

SPI (англ. Serial Peripheral Interface) – Серійний периферійний інтерфейс.

KMF (англ. Kevoree Modeling Framework) – проєкт з відкритим кодом, який спрямований на розробку розподілених систем, що переналаштовуються.

ALM (англ. Application Lifecycle Management) – Управління життєвим циклом додатків.

PLM (англ. Product Lifecycle Management) – Управління життєвим циклом продукції.

ЗМІСТ

ВСТУП

Актуальність теми. Кіберфізичні системи (КФС) пропонують нові способи взаємодії людей з обчислювальними системами: кожна річ тепер інтегрована в обчислювальну потужність, яку можна використовувати для забезпечення безпеки, допомоги, керівництва або просто комфорту для користувачів. CPS - це довговічні та поширені системи, які інтенсивно покладаються на мікроконтролери та малопотужні процесори, інтегровані в будівлі (наприклад, автоматизація для підвищення комфорту та оптимізації енергоспоживання) або автомобілі (наприклад, передові функції безпеки, що включають зв'язок між автомобілями для уникнення зіткнень). CPS працюють у нестабільному середовищі, де вузли повинні співпрацювати в опортуністичний спосіб і динамічно адаптуватися до свого контексту. У цій роботі запропоновано μ-Kevoree, проекцію Kevoree (компонентної моделі, заснованої на models@runtime) на мікроконтролери. μ-Kevoree переносить проблеми динамічності та еластичності безпосередньо в пристрої з обмеженими ресурсами. Його оцінка щодо ключових критеріїв у вбудованій області (використання пам'яті, надійність і продуктивність) показує, що, незважаючи на обмежені накладні витрати, μ-Kevoree забезпечує переваги динамічно реконфігурованої компонентної моделі (безпечна, дрібнозерниста і ефективна реконфігурація) у порівнянні з традиційними методами динамічного оновлення прошивки. Тому розроблення моделі динамічних мереж міської інфраструктури з використанням кіберфізичних системє актуальним напрямком сучасних наукових досліджень.

Мета і задачі дослідження. Метою даної кваліфікаційної роботи освітнього рівня «Магістр» є підвищення рівня повноти подання інформації щодо моделювання динамічних мереж міської інфраструктури з використанням кіберфізичних систем. Для досягнення поставленої мети потрібно виконати ряд завдань, зокрема:

– Проаналізувати стан досліджень в області кіберфізичних систем;

– Дослідити існуючі на даний час методи та інстурменти моделювання кіберфізичних систем;

– Проаналізувати безшовну динамічну адаптацію мікроконтролерів;

– Виконати аналіз існуючих компонентних моделей;

– Розробити модель динамічних мереж міської інфраструктури з урахуванням різноманітних компонентів.

Об’єкт дослідження динамічні мережі міської інфраструктури.

Предмет дослідження. моделі та технології моделювання, використання кіберфізичних систем у контексті управління та оптимізації міської інфраструктури.

Наукова новизна одержаних результатів кваліфікаційної роботи полягає у тому, що отримав подальший розвиток моделювання динамічних мереж міської інфраструктури з використанням кіберфізичних систем.

Практичне значення одержаних результатів. Виконано макетування та моделювання динамічних мереж міської інфраструктури з використанням кіберфізичних систем.

Апробація результатів магістерської роботи. Основні результати проведених досліджень обговорювались на IX науково-технічної конференції «Інформаційні моделі, системи та технології» Тернопільського національного технічного університету імені Івана Пулюя (м. Тернопіль, 2023 р.).

Публікації. Основні результати кваліфікаційної роботи опубліковано у двох працях конференції (Див. додатки А).

Структура й обсяг кваліфікаційної роботи. Кваліфікаційна робота складається зі вступу, чотирьох розділів, висновків, списку літератури з 79 найменувань та 1 додатку. Загальний обсяг кваліфікаційної роботи складає 78 сторінки, з них 72 сторінки основного тексту, який містить 12 рисунків та 3 таблиці.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

Основними рушійними силами розвитку та еволюції кіберфізичних систем (КФС) є скорочення витрат і часу на розробку, а також вдосконалення розроблених продуктів. Це передбачає віртуалізацію продукту для покращення його дизайну, підтримки верифікації та валідації, а також покращення його виробництва та експлуатації. Загалом, віртуалізація забезпечує більшу гнучкість при менших витратах на різних етапах розробки. Взаємодія між розробленим продуктом і виробничими системами відіграє важливу роль у розвитку концепції Індустрії 4.0 (яку також називають передовим виробництвом, розумним виробництвом і кіберфізичними виробничими системами). Майбутні тенденції, методи і моделі процесу проєктування систем також повинні розглядатися з точки зору їхньої ролі в трансформації традиційних системних парадигм, таких як мехатроніка, вбудований інтелект і системи автоматизації, в кіберфізичні системи або глобальну інтеграцію, пов'язану з Інтернетом речей (IoT).

Концепція автоматизації, яка є розвиненою концепцією механізації, почала розвиватися в середині 1950-х років, оскільки зростаюча здатність застосовувати розробки в галузі приладобудування та комп'ютерних технологій для управління системами передбачала перехід до середовища, в якому комп'ютери будуть нести все більшу відповідальність за функціонування цілого ряду процесів. Початкові концепції автоматизації, як правило, асоціювалися з виробничими процесами і можливістю усунути людей від багатьох з них, що призвело до створення безпечнішого середовища, яке виробляло б продукцію з більшою ефективністю і стабільністю. Як, наприклад, заміна електронних систем на процесори, такі як серії PDP-8 і PDP-11, починаючи з середини 1960-х років.

З моменту розробки початкових концепцій автоматизації виробництва комп'ютери все більше впроваджувалися в промисловість для виконання завдань, що охоплюють весь процес проєктування, розробки та виробництва, а також для виконання супутніх завдань, таких як фінансовий контроль, маркетинг і логістика. Крім того, роль промислових обчислень вийшла за межі своєї первісної сфери виробництва і поширилася на такі галузі, як охорона здоров'я, де

здатність збирати і управляти великими обсягами даних оцінюється як така, що має потенційну цінність у мільярди доларів [1]. Однак у багатьох випадках очікувані результати, можливо, ще не матеріалізувалися.

Розглядаючи зростання кількості комп'ютерів, необхідно враховувати низку факторів і проблем, які слід брати до уваги при створенні систем наступного покоління. Зокрема:

1. Перехід від економіки, заснованої на товарах, до економіки, заснованої на інформації або знаннях, інтегрованої з розробками на системному рівні, такими як кіберфізичні системи та Інтернет речей. У таких економіках можливість доступу, передачі і обміну відповідними знаннями на основі контексту і потреб, ймовірно, змінить природу систем, особливо коли вони пов'язані з технологіями "виробництва на вимогу", такими як 3D-друк (адитивне виробництво).

2. Переорієнтація і, можливо, переосмислення ролі користувача в широкому спектрі галузевих видів діяльності, в яких диференціація завдань краще враховує асоціативну і спільну природу діяльності. Наприклад, накопичуються докази того, що поміщення людини з її короткою тривалістю уваги в середовище, де домінують і контролюють комп'ютери, призводить до втрати навичок і зниження здатності до прийняття рішень, а також до зниження здатності до інновацій.

3. Переосмислення процесів проєктування з урахуванням зростаючих рівнів складності систем і нездатності окремих осіб, або навіть груп осіб, належним чином зрозуміти природу системи або її потенційні режими збоїв.

1.1 Еволюція кіберфізичних систем

У цьому розділі представлено огляд різних типів систем та процесу переходу від мехатроніки до систем КФС та хмарних систем (IoT). Мехатроніку можна вважати міждисциплінарною галуззю інженерної науки, метою якої є інтеграція та взаємозв'язок між машинобудуванням, електротехнікою/електронікою, технікою управління та комп'ютерними науками

(які також часто називають інформаційними технологіями) таким чином, щоб їх взаємодія формувала основу для проєктування цілого ряду продуктів і типів продуктів [2]. Взаємодія між розробниками продуктів з різних галузей часто ускладнюється недостатнім взаєморозумінням між дисциплінами та відсутністю спільних платформ для моделювання складних систем.

Оскільки багато підсистем постачаються зовнішніми постачальниками, існує потреба як у горизонтальній інтеграції всередині організацій, так і у вертикальній інтеграції між постачальниками підсистем і постачальниками повних систем. У такому контексті в [3] визначено КФС як «інтеграцію обчислювальних і фізичних процесів». Вбудовані комп'ютери та мережі відстежують і контролюють фізичні процеси, як правило, через петлі зворотного зв'язку, в яких фізичні процеси впливають на обчислення і навпаки.

На рисунку 1.1 показано процес переходу від механічної системи до КФС та Інтернету речей, де горизонтальна вісь відображає рушійні сили/необхідні технології, а вертикальна вісь - рівень системи.

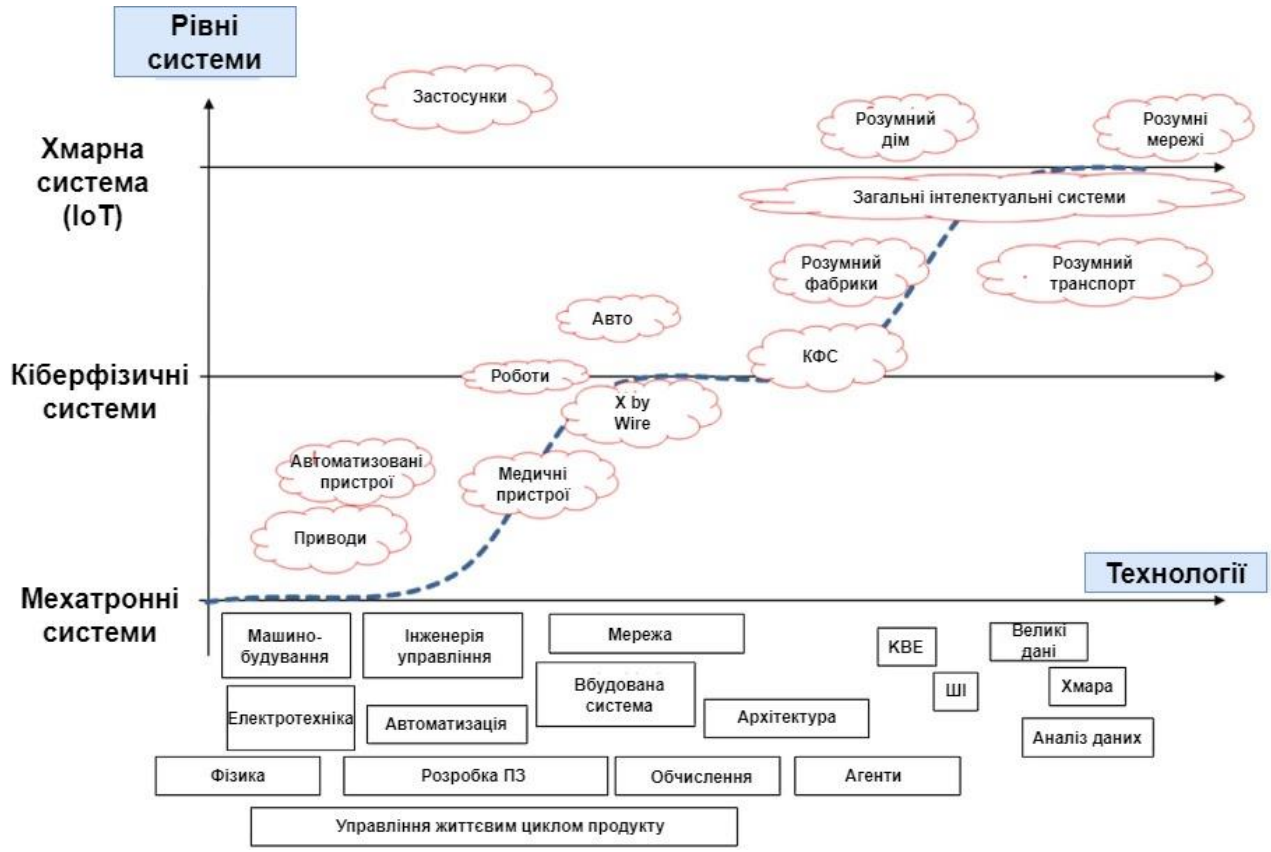

Рисунок 1.1 – Процес переходу від мехатроніки до КФС та Інтернету речей.

1.2 Парадигма КФС

Раніше зміни парадигм, пов'язаних з розвитком комп'ютерних технологій, були не лише рушійною силою еволюції, але й спричинили революцію в системних концепціях і мисленні [4]. Обіцянка кіберфізичної революції полягає в тому, що вона дозволяє відстежувати, контролювати і впливати на фізичний світ як адаптивно, так і інтелектуально. Багато дослідників вважають, що парадигма кіберфізичних обчислень матиме далекосяжні наслідки у сфері інженерних систем. Однак розвиток КФС переплітається з технологічними, економічними та соціальними можливостями і викликами. Тому існує потреба в нових трансдисциплінарних теоріях, моделях, методах, інструментах і контекстах для підтримки цих розробок.

Досягнення цієї здатності також викликає потребу у спільній семантиці і концептуальних рамках для створення мостів між механікою, електронікою, інженерією, управлінням і обчислювальною технікою [5]. Це важливо з огляду на історичний розвиток сучасної ситуації, коли онтологія, епістемологія, методологія, аксіологія і праксеологія інформаційних та обчислювальних наук, з одного боку, і фізичних наук (природничих, хімічних, біологічних), з іншого, розділені, що призводить до розбіжностей у наукових засадах і технологіях.

У той час як мережеві персональні комп'ютери, вбудовані, повсюдні можемо вважати, що вони ведуть до функціональної і реалізаційної диверсифікації типів технічних систем, кіберфізичні обчислення стимулюють технологічну гомогенізацію і конвергенцію в системних проявах [6]. У цьому контексті нині диференційовані системи, ймовірно, демонструватимуть все більше схожості з точки зору технологій та функціонального спектру, і ця конвергенція, ймовірно, стане все більш значущою в найближчому майбутньому. Ми вже є свідками сильного зближення кіберсвіту (створеного людськими знаннями і цифровими обчисленнями) і фізичного світу (наприклад, у вигляді сконструйованих технічних систем).

З цих причин кіберфізичні системи мають не лише академічну перспективу, але й урядову.

Наприклад, програма Європейського Союзу "Передові дослідження і технології для вбудованого інтелекту і систем" (ARTEMIS) інвестувала сім мільярдів євро в дослідження і розробки в період з 2007 по 2013 рік, щоб стимулювати роботу над досягненням лідерства в галузі інтелектуальних електронних систем до 2016 року [7]. Національний науковий фонд (NSF) США і рамкова програма ЄС "Горизонт 2020" сформулювали і профінансували проєкти, що працюють над теоріями, технологіями і реалізацією XIC з 2007 і 2010 років, відповідно [8].

Крім того, багато великих і міжнародних компаній розглядають КФС як парадигму для своїх майбутніх систем і послуг. У США консорціум, до якого входять Каліфорнійський університет в Берклі та Каліфорнійський технологічний інститут, а також кілька компаній-партнерів, створив Центр промислових кіберфізичних систем (iCyPhy). В Європі було створено державно-приватне партнерство під назвою "Електронні компоненти і системи для європейського лідерства" (ECSEL) [8].

1.3 Інтернет речей

Концепція Інтернету речей як середовища для обміну інформацією між великою кількістю взаємопов'язаних пристроїв і систем, що лежить в основі цієї концепції, сягає своїм корінням далеко за межі того часу, коли ця назва була вперше вигадана. В останні роки розвиток цієї концепції був зумовлений зростанням доступності розумних пристроїв, які можуть підключатися до Інтернету і, часто автономно, обмінюватися інформацією між собою. Це, в свою чергу, призвело до розвитку нових системних концепцій, в яких такі смарт-пристрої інтегруються через доступ до інтернет-ресурсів (програмне забезпечення як послуга (SaaS), платформа як послуга (PaaS) та інфраструктура як послуга (IaaS)) для створення контекстно-орієнтованих систем на вимогу. Крім того, це створює середовище, в якому інформація може розглядатися як товар, цінність якого для системи, а отже і для користувача, значною мірою визначається контекстом.

Поєднання цього середовища, заснованого на інформації, а отже, на знаннях, зі здатністю аналізувати великі обсяги даних створює нові бізнес-моделі, водночас породжуючи проблеми конфіденційності, управління та доступу до індивідуальних даних [10,11].

У сукупності вищезазначене ставить перед розробниками систем низку викликів, в тому числі такі аспекти, як:

Природа інформації:

- Історично склалося так, що інформація використовувалася для об'єднання артефактів у систему.
- У контексті Інтернету речей і, власне, кіберфізичних систем, "розумні" артефакти використовуються для обслуговування інформації для створення системи.
- Необхідна інформація може бути локальною або віддаленою і пов'язана з потребами користувача.
- Це вимагає систем, які розуміють контекст, щоб шукати інформацію, пов'язану з поточними потребами.
- Інтелектуальні модулі можуть додаватися або відніматися в міру необхідності як частина континууму інформації та об'єктів.
- Системи домовляються про інформацію на основі контексту.
- Інформація як товар.
- Цінність визначається вимогами та потребами користувача.

Невизначеність у проєктуванні, а саме – системи на основі Інтернету речей все більше самоорганізуються і залежать від контексту, що призводить до виникнення ряду проблем, серед яких:

- Необхідність забезпечення ефективної комунікації між членами команди проєктувальників та засоби досягнення цього.
- Розробка та впровадження інтелектуальних, адаптивних та орієнтованих на користувача інтерфейсів.
- Моделювання абстракцій, особливо під час проєктування. Ефективна інтеграція інформаційних та фізичних моделей для координації дій у реальному часі між обома середовищами.

- Прототипування та тестування в інформаційному середовищі.
- Безпека даних та безпека доступу.
- Соціальні питання, пов'язані з управлінням та використанням даних, зокрема персональних даних.
- Досягнення конфіденційності за задумом за наявності невизначених системних структур.

Концепції Інтернету речей та кіберфізичних систем ставлять перед розробниками виклик реалізації структур в інформаційно насиченому середовищі, де інформація та комунікації все більше стають рушійною силою процесу проєктування. Це, в свою чергу, вимагає від розробників доступу до нових і нових засобів моделювання, здатних представляти такі ситуації. У контексті промисловості здатність генерувати нові продукти і послуги на основі обміну інформацією, що забезпечується Інтернетом речей, ймовірно, буде мати вирішальне значення, так само як і здатність надавати користувачам широкий спектр розумних об'єктів, необхідних їм для підтримки створених систем. Наприклад, багато з цих систем і пов'язаних з ними "розумних" об'єктів не будуть запрограмовані в звичайному розумінні, а матимуть функціональні можливості, визначені користувачами. Актуальність буде визначатися користувачами у різний спосіб.

Це вимагає від інтерфейсу здатності перетворювати наміри користувачів, особливо нетехнічних, у необхідні програмні структури, що визначають роботу системи. Робота системи може, в свою чергу, вимагати ідентифікації ресурсу на основі Інтернету речей, наприклад, для аналізу похідних даних, і це повинно бути зроблено таким чином, щоб система користувача і пов'язані з нею дані користувача не були скомпрометовані. З точки зору промисловості, це означає, що з'явиться набагато більше невизначеності щодо того, як їхній продукт або процес буде використовуватися, і як його потрібно буде інтегрувати з іншими продуктами і системними артефактами від різних постачальників для створення нових систем.

Посилаючись на рисунку 1.2, взаємозв'язок між різними системними рівнями мехатроніки, кіберфізичних систем і Інтернету речей можна виразити в

термінах зростаючих рівнів абстракції, до яких потім необхідно додати користувача.

На рисунку 1.2, на рівні пристроїв (мехатроніки) процеси і процедури проєктування досить добре визначені і зрозумілі і підтримують переклад у фізичні системи. На рівні кіберфізичних систем зв'язок між фізичними компонентами і кіберрівнем значною мірою визначається функціями. Проте, оскільки системи на цьому рівні стають більш складними, здатність людей розуміти їх функціонування зменшується. На рівні Інтернету речей є багато погано визначених або невизначених складових, які будуть невідомі розробнику системи, окрім як з точки зору їхнього бажаного внеску в роботу системи та її функції.

Рисунок 1.2 – Системні рівні абстракції між технологіями

1.4 Висновок до першого розділу

Протягом останніх років було докладено значних зусиль для розуміння взаємозв'язку людини з фізичним середовищем, наприклад, за допомогою таких концепцій дизайну, як «Дизайн для всіх» (Design for All). Тому виникає потреба в подібному підході до дизайну. що пов'язує людину з її інформаційним

середовищем, а також способи, за допомогою яких фізичне та інформаційне середовище можуть бути більш ефективно інтегровані на рівні людини.

Розглядаючи роль комп'ютерів у промисловості, можна припустити, що необхідно ширше поглянути на їхню роль, ніж просто як на контролерів пристроїв, оскільки їхнє використання дедалі більше проникає в усі аспекти промислових систем. Можна стверджувати, що відносні ролі і відносини між людьми і комп'ютерами потребують переоцінки.

2 МЕТОДОЛОГІЇ ДЛЯ ПРОЄКТУВАННЯ КФС

Розробка КФС-дизайну терміново вимагає впровадження високорівневих інтегрованих методів проєктування, в яких проєктувальники розглядають всі інженерні дисципліни одночасно, а також операційні питання, такі як конфіденційність. Ранні етапи проєктування, такі як концептуальне проєктування і системне моделювання, відіграють важливу роль у проєктуванні і розглядаються як сприятливі основи для остаточної форми системи [12,13]. Для більшої наочності, звертаючись до рисунок 3, міждисциплінарна інтеграція розглядається з точки зору як просторового виміру (дисципліни, представлені горизонтальною віссю), так і часового виміру (процес проєктування, представлений вертикальною віссю).

2.1 Основні виклики при розробці та інтеграції КФС

КФС потребує мультидисциплінарного процесу розробки [14], під час якого розробники повинні зосередитися не лише на окремих фізичних та обчислювальних компонентах, але й на їх інтеграції та взаємодії. Тому проблеми, пов'язані з КФС-дизайном, розглядаються тут з точки зору фізичного процесу, обчислень та інтеграції відповідно.

З точки зору фізичного процесу, КФС безпосередньо пов'язані з фізичним світом. Вони виявляють зміни в цьому світі та реалізують управління поведінкою системи в реальному часі. Порівняно з іншими більш традиційними формами складних систем (наприклад, мехатронними системами), вплив навколишнього середовища (тобто відхилення і невизначеності в навколишньому середовищі або створені людиною) відіграє значну роль для КФС [15].

З точки зору обчислень, вважається, що КФС розробляються на основі вбудованих систем [16]. Однак, зв'язок з фізичним світом КФС набагато сильніший, ніж у вбудованих систем. Тому програмна складова КФС набагато більша і складніша, ніж у вбудованих систем. З точки зору інтеграції обчислювальних і фізичних процесів, проєктування КФС вимагає інтеграції в

рамках процесу спільного проєктування КФС. Фізичні та обчислювальні компоненти взаємодіють один з одним між дисциплінами, що призводить до переривань і несумісності в процесі проєктування. Також часто бракує чітко визначених і задокументованих взаємодій та інтерфейсів між різними дисциплінами та залученими компонентами, що ускладнює взаєморозуміння у спілкуванні [17].

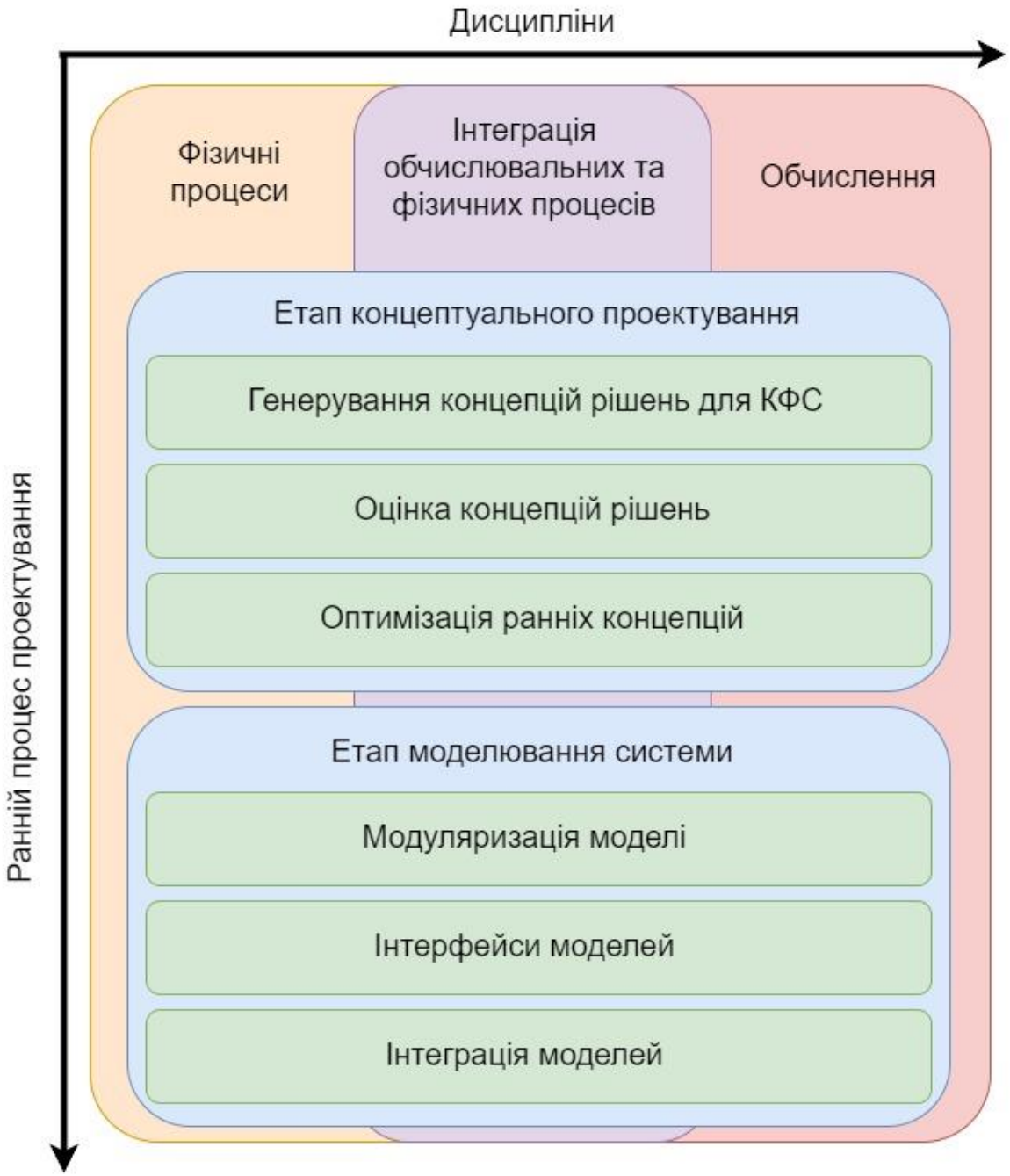

Рисунок 2.1 – Дисциплінарна інтеграція на ранніх стадіях проєктування КФС.

2.2 Функціональний та архітектурний дизайн КФС

Методологічну підтримку дозволяє впоратися зі складністю проєктування КФС на ранніх етапах проєктування. Це, в свою чергу, безпосередньо пов'язано з функціональним та архітектурним дизайном КФС. З точки зору етапу концептуального проєктування, характерні для проєктування аспекти систем, такі як параметри ієрархії та модульність, повинні бути проаналізовані за допомогою концептуальної моделі [18]. На завершення етапу концептуального проєктування повинні бути запропоновані концепції рішень для проєктування КФС. На етапі концептуального проєктування існують три основні проблеми.

Виникає задача генерування концепції рішень на основі вимог замовника або результатів дослідження ринку. Хоча для структурування вимог замовника були запропоновані різні інструменти та мови, процес переходу від специфікації вимог до концепції рішення повинен бути визначений. Різні концепції рішень можуть бути запропоновані окремими розробниками або проєктними групами відповідно до їхніх знань та професійного досвіду.

Друге завдання полягає в оцінюванні різних запропонованих концепцій та рішень.

Через складність КФС неможливо знайти оптимальні концепції рішень без ітерацій, тому існує вимога оптимізувати концепції рішень на ранній стадії процесу проєктування.

Мова системного моделювання (SysML) була створена Міжнародною радою з системної інженерії (INCOSE) та Групою управління об'єктами (OMG) [91] на основі уніфікованої мови моделювання (UML) для підтримки системної інженерії, заснованої на моделях (MBSE). Для моделювання СЕС були запропоновані й інші мови моделювання. Дослідники в [20] вважають, що неможливо формально описати динамічну КФС через гнучкість, в якій можуть бути визначені неповні діаграми UML, тому вони пропонують мову опису архітектури (ADL) для динамічного моделювання архітектур зі специфічним посиланням на СФС. Високодинамічна та розподілена автономна архітектура може бути описана за допомогою ADL. На відміну від мов моделювання, таких

як UML/SysML або ADL, Modelica може не тільки представляти систему, але модель, створена в Modelica, також може бути перенесена в математичні моделі для імітаційного моделювання. Описано декілька методів проєктування на основі Modelica [21,22,23].

2.3 Методи та інструменти для моделювання та імітації КФС

З точки зору етапу системного моделювання, КФС може бути визначена як сукупність компонентів і відповідних інтерфейсів між ними. Метою етапу системного моделювання є створення міждисциплінарної моделі для визначення важливих функціональних і системних параметрів і, таким чином, реалізація потенційної оптимізації [24]. Створенню міждисциплінарних моделей на етапі моделювання в процесі проєктування КФС перешкоджають три основні проблеми. Згідно з визначенням КФС, моделі КФС зазвичай включають як фізичні, так і обчислювальні компоненти [25], тому першою проблемою є те, як моделювати компоненти різних дисциплін. Друге питання полягає в тому, як моделювати інтерфейси між компонентами, взятими з різних дисциплін, використовуючи спільну термінологію. Останнє питання пов'язане з інтеграцією моделей. Існують усталені методи для моделювання динаміки фізичних компонентів (тобто неперервна модель) і дискретної поведінки обчислювальних компонентів (тобто дискретна модель), але інтерфейсу для об'єднання неперервної та дискретної моделей до цього часу не приділялося значної уваги.

Автори в [26] запропонували часову семантику та її центральну роль у розробці нової системи міркувань на основі часу в рамках MBSE для КФС. Згідно з їхньою пропозицією, онтологічна структура описує поведінку системи в термінах часу, інтервалів часу та взаємозв'язків між інтервалами часу, і тоді може бути забезпечена коректність функціональності щодо вимог. Автори в [27] розробили підхід до моделювання на основі агентів, щоб допомогти проєктувальникам вирішити проблеми, що виникають через фундаментальні відмінності в роботі кібернетичних і фізичних компонентів, а також через значну взаємозалежність між цими компонентами.

Окрім методів проєктування, що базуються на мовах моделювання та онтологічного моделювання, для підтримки КФС-проєктування на основі MBSE було розроблено декілька методів проєктування на основі комерційного мультидисциплінарного програмного забезпечення для моделювання та імітаційного моделювання, такого як MATLAB [28], AMESim [29]. Програмне забезпечення для моделювання надає різні типи бібліотек з різних дисциплін, які допомагають дизайнерам перевіряти, оцінювати та оптимізувати свій дизайн.

2.4 Управління життєвим циклом продукту для КФС

Різні дослідницькі проєкти, пов'язані зі специфікацією підходу PLM для КФС, який підтримує інтеграцію мережевої економіки, зростання Інтернету речей та розвиток мехатронних систем передбачені в [30,31]. Враховуючи появу концепції КФС, існує потреба в розробці моделей продуктів і методів системної інженерії з області мехатроніки в напрямку КФС [32,33]. Для розробки таких складних систем, як КФС, компанії в даний час використовують кілька видів інформаційних систем (ІС) для управління та інтеграції інженерних даних, пов'язаних з кожною з дисциплін, задіяних під час проєктування.

Наприклад, системи Product Data Management (PDM) зазвичай використовуються для управління даними про апаратне забезпечення (HW) [34], тоді як системи Software Configuration Management (SCM) використовуються для управління даними про програмне забезпечення (ПЗ) [35]. PDM, як правило, вважається одним з найважливіших будівельних блоків ІС для реалізації підходу управління життєвим циклом продукту (Product Lifecycle Management, PLM). PLM визначається як систематична концепція для інтегрованого управління всією інформацією, пов'язаною з продуктом і процесом, протягом усього життєвого циклу, від початкової ідеї до кінця життя [36]. Зосереджуючись на основних функціональних можливостях (контроль версій, паралельна розробка, вибір конфігурації та управління робочим простором), SCM дуже схожа на PDM, а управління життєвим циклом додатків (Application Lifecycle Management, ALM) можна порівняти з PLM. Дійсно, ALM "з'явився для позначення

координації діяльності та управління артефактами (наприклад, вимогами, вихідним кодом, тестовими кейсами) протягом життєвого циклу програмного продукту [37]". ALM походить з області управління конфігурацією, яка традиційно забезпечує зберігання, керування версіями та відстеження між артефактами [38]. Воно має додаткову функціональність для підтримки комунікації, звітності, відстеження та інтеграції інструментів, таких як управління вимогами та дефектами або засоби збірки та тестування, як показано на рисунку 2.2.

Навіть якщо ALM і PLM збігаються в деяких аспектах (наприклад, для врахування декількох фаз життєвого циклу), залишаються проблеми в управлінні інженерними даними під час проєктування мехатронних систем або КФС. Наприклад, як показано в [39], відкритим питанням залишається створення єдиної специфікації матеріалів (BOM), яка об'єднує всі компоненти HW і SW, а також визначення загальної моделі даних, яка керує всією системою на кожному етапі проєктування. Така унікальна специфікація необхідна для відстеження змін у проєкті та виявлення пов'язаних з ними впливів, особливо між різними дисциплінами. Але мати унікальну специфікацію, розділити механічні частини, електронні/електричні компоненти та програмні коди недостатньо. Автори [40] пояснюють, що одне і те ж авіаційне обладнання (тобто підсистема) може використовуватися з різним програмним забезпеченням на різних літаках. У такому випадку обладнанням і програмним забезпеченням потрібно управляти відповідно.

Для того, щоб проєктні зміни, виконані на обладнанні або в програмному забезпеченні, не впливали на глобальну узгодженість системи, функції управління версіями і вибору конфігурації ALM і PLM повинні бути адаптовані до обраної структури продукту. Цей висновок призводить до іншого важливого питання щодо інтероперабельності інформаційних систем та інженерних додатків [41,42,43]. Інтеграція послуг у розробку продукту і тісна інтеграція розробки продукту і розробки виробничих систем (CPPS) обговорюється в роботі [44]. Крім того, міждисциплінарний характер таких систем і потенційні неузгодженості на всіх етапах проєктування запропоновані авторами в [45].

Рисунок 2.2 – Основні елементи управління життєвим циклом додатків.

2.5 Архітектура та поведінкова парадигма для кіберфізичних систем

Тенденція до створення КФС з більшою кількістю давачів, більшою кількістю виконавчих механізмів і більшою кількістю зв'язків не зупиниться. Майбутня КФС може складатися буквально з мільйонів давачів і виконавчих механізмів, керованих тисячами процесорів. Наявність такої кількості активних і пасивних елементів у постійно мінливому середовищі (чим більшою стає система, тим більшою стає територія, яку вона охоплює, а отже, зростає кількість змін і флуктуацій). Простий розрахунок показує, що надійність всієї системи стане єдиною найбільш загрозливою проблемою [46]. Це означає, що система повинна гнучко адаптуватися до постійно мінливого зовнішнього середовища (наприклад, до збоїв), а також до внутрішніх умов (наприклад, до збоїв). Крім того, майбутні СУП повинні працювати автономно, тобто без втручання людини. Зокрема, збої в роботі системи повинні діагностуватися та усуватися без негайного втручання людини. Концепція відмовостійкої архітектури вважається перспективною з цих причин.

Мінімальна архітектура КФС складається з трьох елементів давачів, виконавчих механізмів та інтелекту (контролерів), що відрізняє КФС від мехатронної системи - це інтелектуальний взаємозв'язок та обмін інформацією між пристроями. Архітектура системи визначає, як давачі та виконавчі механізми організовані та поводяться у відповідь на зовнішнє середовище та внутрішні умови, визначені контролерами (ISO/IEC/IEEE42010 [47]). На рисунку 2.3 зображено просту сучасну архітектуру КФС. У той час як всередині машини давачі і виконавчі механізми з'єднані через шину (наприклад, Profibus) і управляються контролерами, ці машини і роботи в подальшому будуть підключені до мережі (наприклад, Profinet, так зване мережеве з'єднання на рисунку 2.3 - зелені стрілки) і управлятися контролером вищого рівня. Очевидно, що така архітектура дозволяє реконфігурацію на рівні мережі, але не на рівні машини.

На противагу цьому, майбутні КФС, побудовані з відмовостійкою архітектурою, повинні дозволяти як статичну, так і динамічну реконфігурацію на будь-якому рівні [48]. Така стійка архітектура може бути досягнута шляхом віртуалізації, що означає, що логічна функція, наприклад, сенсорної мережі, буде віртуально визначена і відокремлена від фізичної мережі [49]. Якщо в мережі давачів виявлено будь-яку несправність або якщо змінюються цілі зондування, давачі будуть логічно переконфігуровані за допомогою міркувань на архітектурному рівні. Аналогічно, несправності виконавчих механізмів та контролерів можуть бути виявлені та усунені шляхом функціональної реконфігурації.

На рисунку 2.4 показано приклад реалізації такої відмовостійкої архітектури КФС, яка дозволяє логічну реконфігурацію будь-якого елементу. Мережі давачів і мережі виконавчих механізмів будуть віртуалізовані. Таким чином, щоразу, коли змінюється призначення системи, зовнішнє середовище або внутрішні умови, вся система зможе оптимально реконфігурувати свої давачі і виконавчі механізми на основі міркувань інтегрованого інтелекту (контролера). Наприклад, на рисунку 2.4 контролер 1 використовує давач 1 для керування приводом 1. Припустимо, що привід 1 виявився несправним. У такому випадку

можна задіяти привід 2 з аналогічною функціональністю за допомогою давача 2 (замість давача 1). Така "реконфігурація" сприяє підвищенню стійкості всієї системи.

Хоча сенсорна мережа може включати в себе надмірність для підвищення надійності, це може збільшити невизначеність сенсорних даних [50,51]. Аналогічно, віртуальна сенсорна мережа, що складається з давачів, які спочатку не призначені для вимірювання відповідної фізичної величини, також може вносити неточність і невизначеність. Подібно до давачів, резервні приводи можуть бути корисними для підтримки функціональної реконфігурації мережі приводів, коли деякі приводи виходять з ладу або виконують різні функції [52]. Наприклад, електромобілем, оснащеним колісним двигуном для кожного окремого колеса (тобто чотирма двигунами замість одного), можна керувати, контролюючи швидкість кожного з чотирьох коліс окремо. Ці двигуни також можуть замінити механічні гальма (за допомогою рекуперації та створення протилежного обертального моменту), а це означає, що тепер доступне "віртуальне гальмо", яке може замінити механічне гальмо. Такий підхід до архітектурного проєктування віртуальних мереж приводів на системному рівні має потенціал не тільки забезпечити кращу відмовостійкість, але й створити абсолютно нове та інноваційне рішення. Динамічна реконфігурація та відмовостійкість віртуальних приводів є ключем до досягнення відмовостійкої поведінки системи.

Мережі віртуальних давачів і виконавчих механізмів повинні розроблятися та реконфігурувати за допомогою системних міркувань про фізичні давачі та виконавчі механізми. Ці можливості міркувань відіграють центральну роль у всій архітектурі системи, щоб вона стала стійкою до зовнішніх змін і збоїв.

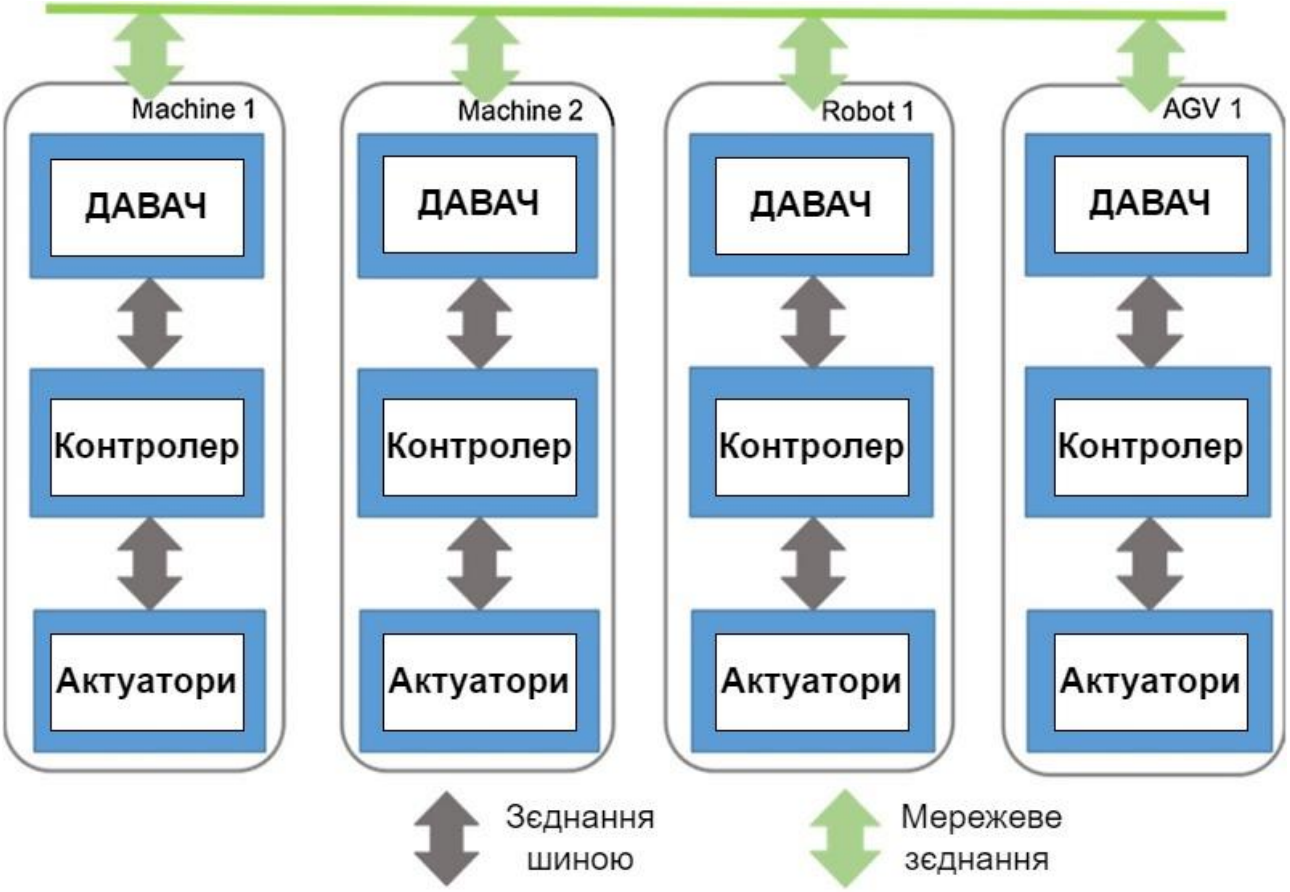

Рисунок 2.3 – Поточна архітектура КФС.

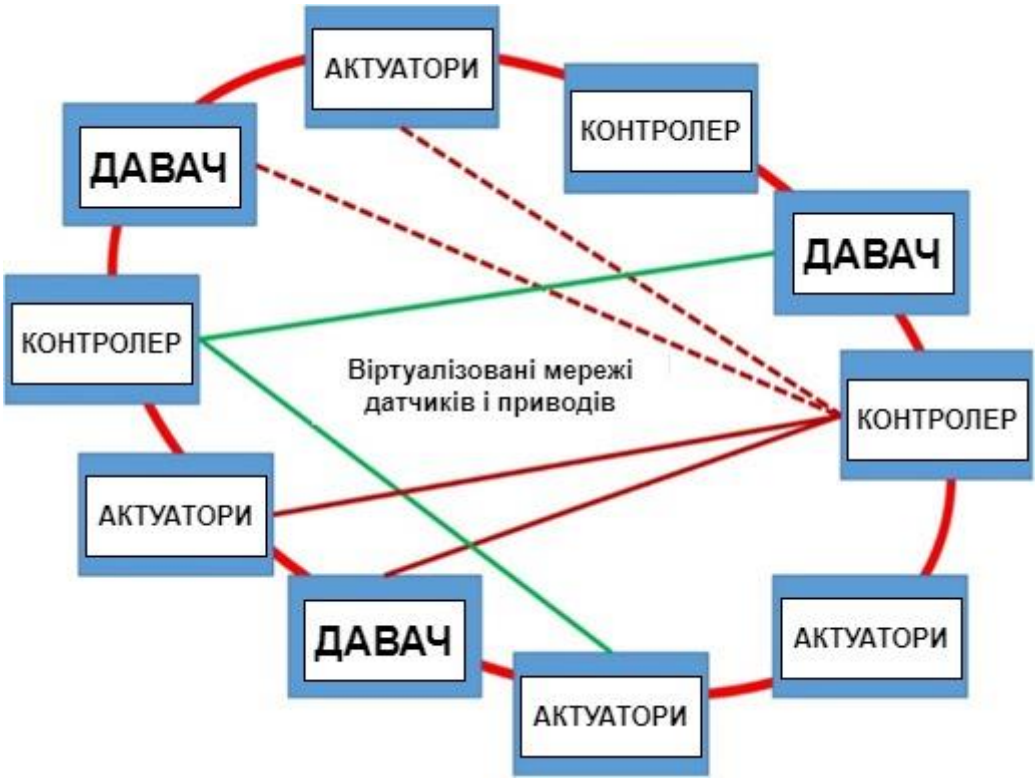

Рисунок 2.4 – Відмовостійка архітектура КФС.

В останні роки агенти та агентна орієнтація вийшли на рівень нової архітектури та парадигми поведінки. Спочатку з'явившись зі сфери штучного інтелекту [53], вони були інтегровані в широкі сфери управління виробничими системами [54,55,]. Агента в контексті автоматизації виробництва можна розглядати як певну сутність, призначену для досягнення своїх цілей самостійно, але спільно, взаємодіючи при цьому з навколишнім середовищем, яке може включати інших агентів.

Агентно-орієнтований підхід став придатним і ефективним засобом для вирішення складних проблем в рамках інжинірингу та систем управління виробничими системами. Цей підхід відводить реальним та існуючим суб'єктам виробничої системи центральну роль і, крім того, явно враховує цілі та завдання виробництва, а також необхідну поведінку для їх досягнення. Таким чином, агент-орієнтований підхід особливо застосовний до класів проблем зі складними і дуже мінливими граничними умовами, а також з проблемою самоадаптації. Прикладами таких класів систем є системи планування виробництва, логістичні системи або системи технічного обслуговування [56].

Як зазначалося вище, агенти можуть бути реалізовані на ПЛК та ПК залежно від вимог реального часу та необхідної обчислювальної потужності. Якщо потрібен реальний час, то обирають ПЛК [57]. Надійність гарантується за допомогою оболонки безпеки, яка дозволяє агенту приймати лише безпечні та неруйнівні рішення [58] під час виконання. Натомість знання агентів моделюються [59] під час проєктування у вигляді діаграми параметрів, але також можуть бути засвоєні під час виконання. Приклад поведінки невеликої лабораторної установки, що навчається [60]. Крім того, агенти корисні для виявлення закономірностей в роботі, наприклад, аварійних сигналів, щоб значно зменшити кількість повідомлень.

Агент управління фільтрує та агрегує повідомлення, формуючи базу знань агента на скінченному автоматі [61]. Дискретні потоки повідомлень, що надходять на станцію, аналізуються, а фільтри і класифікатори використовуються для подальшого зменшення кількості відповідних послідовностей повідомлень і визначення першопричини аномальної ситуації.

Нарешті, виключно критичні ситуації, тобто критичні послідовності повідомлень, групуються і візуалізуються для оператора, щоб полегшити йому реагування на тривоги, значно зменшивши кількість повідомлень, а отже, і пов'язані з ними управлінські зусилля.

Підхід ґрунтується на концепції частотного аналізу журналів промислових повідомлень для виявлення послідовностей, що складаються з причинно-наслідкових зв'язків між повідомленнями. Алгоритм, який реалізує запропоновану концепцію, складається з двох частин. У першій частині він застосовує абстрактні стратегії розпізнавання на основі критеріїв (RS), призначені для комбінування тегів, які зазвичай використовуються в системах управління тривогами. Вони не визначають шаблони сповіщень, які потрібно ідентифікувати, натомість вони визначають стратегії пошуку корельованих послідовностей на основі абстрактних апріорних знань, беручи до уваги налаштовані теги тривог. У другій частині алгоритм частотного пошуку шаблонів аналізує журнали повідомлень для виявлення часто повторюваних шаблонів при застосуванні ПК до ідентифікованих шаблонів. Застосовуючи реалізоване розпізнавання шаблонів, алгоритм здатний генерувати формальні правила, які можуть бути переведені в графічне представлення для перегляду виявлених послідовностей, придатних для читання людиною. Це значно підвищує зрозумілість підпослідовностей, що з'являються, для користувачів-людей.

Критично важливим компонентом є база знань агента. Агент повинен надійно візуалізувати всі критичні ситуації, обмежуючи при цьому фільтрацію та агрегування повідомлень. Підхід машинного навчання за принципом "людина в циклі" було обрано для того, щоб уможливити автоматичне виявлення послідовностей повідомлень під наглядом людини. Історичні журнали тривог використовуються для виявлення значущих послідовностей повідомлень на основі статистичних методів розпізнавання образів для придушення надлишкових повідомлень і візуалізації критичних ситуацій [61]. Застосування машинного навчання до історичних даних тривог призведе до послідовностей сповіщень, які є статистично правильними, але логічно безглуздими [62]. Для

того, щоб зменшити зусилля оператора, застосовується попередня обробка на основі критеріїв, що ґрунтується на фонових знаннях, таких як умовні позначення та документація заводу. Крім того, застосовується постобробка на основі тестів на значущість, щоб ще більше зменшити кількість шаблонів, які повинен переглядати оператор.

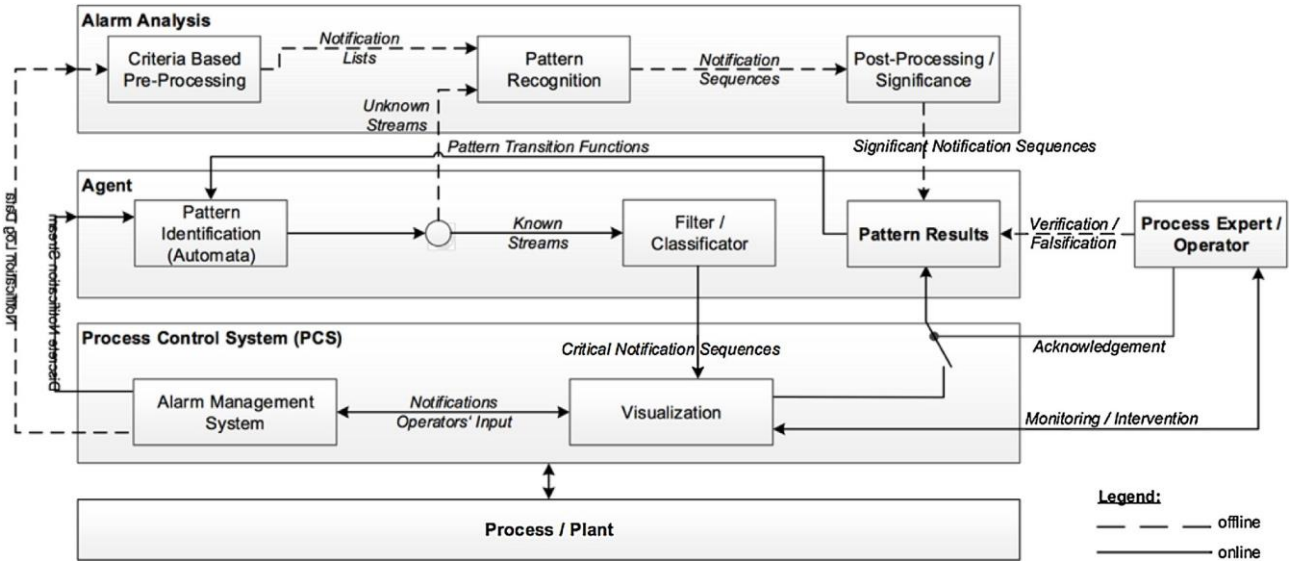

Рисунок 2.5 – Блоки для інтелектуального, динамічного управління тривогами з використанням агентно-орієнтованої ідентифікації шаблонів [79].

Крім деяких досліджень і розробок, необхідні можливості роботи в реальному часі, необхідні можливості агентної технології в режимі реального часу, які обмежують застосування агентної технології, все ще залишаються відкритою проблемою [63]. Тим не менш, за межами агентної технології як програмної технології, агентна орієнтація як парадигма моделювання та інжинірингу в даний час цілком застосовна. Це особливо актуально для випадків використання, що мають справу з обмеженими жорсткими умовами реального часу, наприклад, у випадках контролю виконання виробничих процесів, технічного обслуговування, розподілу даних і документування.

2.6 Висновок до другого розділу

У висновку, розробка кіберфізичних систем вимагає мультидисциплінарного підходу, де розробники фокусуються не лише на ізольованих фізичних та обчислювальних компонентах, але й на їх інтеграції та взаємодії. Проблеми, пов'язані з дизайном КФС, розглядаються через призму фізичного процесу, обчислень та необхідної інтеграції. З точки зору фізичного процесу, КФС прямо взаємодіють з фізичним світом, реагуючи на зміни та керуючи системою в реальному часі. Порівняно з іншими складними системами, вплив навколишнього середовища має значення для КФС. З точки зору обчислень, КФС базуються на вбудованих системах, проте їх програмна складова виявляється значно більшою та складнішою. У контексті інтеграції обчислювальних і фізичних процесів проєктування КФС вимагає спільного підходу та взаємодії між різними дисциплінами, що може призводити до труднощів у взаєморозумінні та комунікації через відсутність чітко визначених і задокументованих взаємодій та інтерфейсів.

3 ДИНАМІЧНА КОМПОНЕНТНА МОДЕЛЬ ДЛЯ КІБЕРФІЗИЧНИХ СИСТЕМ

3.1 Динамічна компонентна модель Kevoree

Kevoree - це динамічна компонентна модель з відкритим вихідним кодом, яка покладається на моделі під час виконання [64] для належної підтримки динамічної адаптації розподілених систем. Models@runtime по суті просуває ідею рефлексії [65] на крок далі, розглядаючи рівень рефлексії як реальну модель, яку можна від'єднати від працюючої архітектури (наприклад, для міркувань, валідації та симуляції) і пізніше автоматично ресинхронізувати з її працюючим екземпляром.

У Kevoree реалізовано models@runtime [66] далі. Зокрема, Kevoree забезпечує належну підтримку розподілених models@runtime. З цією метою вводиться концепція Node для моделювання топології інфраструктури та концепцію Group для моделювання семантики міжвузлової комунікації під час синхронізації моделі відображення між вузлами. Kevoree включає в себе концепцію Channel, яка дозволяє реалізувати декілька семантик зв'язку між віддаленими компонентами, розгорнутими на гетерогенних вузлах. Всі концепції Kevoree (Компонент, Канал, Вузол, Група) підпорядковуються шаблону проєктування типів об'єктів [67] для відокремлення артефактів розгортання від артефактів виконання. Kevoree підтримує різні типи технологій виконавчих вузлів (наприклад, Java, Android, MiniCloud, FreeBSD, Arduino).

Kevoree має на меті надати розширені можливості адаптації для різних типів вузлів:

- Рівень 1: Параметрична адаптація. Динамічне оновлення значень параметрів, наприклад, зміна частоти дискретизації в компоненті, що обгортає фізичний сенсор (адаптація властивостей екземпляра).

- Рівень 2: Архітектурна адаптація. Динамічне додавання або видалення прив'язок або компонентів, наприклад, реплікація програмних компонентів і

каналів на різних вузлах для балансування навантаження (адаптація графа екземплярів).

- Рівень 3: Динамічне резервування типів. Гаряче розгортання типів компонентів, які не були передбачені до початкового розгортання системи. Це дозволяє розвивати систему, забезпечуючи параметричні та архітектурні реконфігурації, включаючи управління екземплярами для типів, які додаються та управляються динамічно (адаптація типів).

- Рівень 4: Адаптація для віддаленого управління. Вузли, що підтримують адаптацію рівня 4, беруть участь у рівні віддаленого управління, який контролює менш потужні вузли. Цей рівень контролює віддалені вузли, запитуючи їх поточну модель Kevoree; рівень запускає динамічну адаптацію вузлів, надсилаючи їм заздалегідь обчислені сценарії реконфігурації. Цей віддалений процес адаптації підтримує безперешкодне управління менш потужними вузлами більш потужними, які мають достатньо ресурсів для побудови та оцінки нових і відповідних конфігурацій.

Механізм адаптації спирається на порівняння моделей між двома моделями Kevoree для обчислення сценарію безпечної реконфігурації системи; виконання цього сценарію переводить систему з поточної конфігурації в нову обрану конфігурацію [68]. Порівняння моделей дає дельта-модель, що визначає зміни (за допомогою CRUD-операцій), які слід застосувати до вихідної моделі, щоб отримати цільову модель. Алгоритми планування [69] використовують цю дельта-модель як вхідні дані для визначення ефективного розкладу кроків адаптації. Дельта-модель остаточно компілюється у скрипт Kevoree. Мова сценаріїв Kevoree Script (скорочено KevScript) є основною мовою для опису реконфігурації. KevScript можна порівняти з FScript для фрактальної компонентної моделі [70]. Виконання KevScript безпосередньо адаптує систему Kevoree, без необхідності повного визначення моделі Kevoree. Такі адаптаційні скрипти пишуть дизайнери, або вони можуть бути згенеровані автоматизованими процесами (наприклад, у контурі керування системою Kevoree).

3.2 Концепція kevoree для мікроконтролерів

Динамічна адаптація є ключовою концепцією для побудови передових КФС, здатних адаптуватися до контексту та потреб користувачів. Моделювання під час виконання є ефективним підходом до управління складністю динамічної адаптації [71] шляхом забезпечення контролю та абстрагування механізмів рефлексії. Застосування методів рефлексії є досить простим на повноцінних моделях компонентів або сервісів, таких як OSGi, або безпосередньо на сучасних об'єктно-орієнтованих мовах, таких як Java, за умови, що апаратне забезпечення виконання є достатньо потужним, щоб запустити віртуальну машину. Вбудоване програмне забезпечення, яке відчуває і діє на фізичний світ (через апаратні компоненти), повинно мати можливість адаптуватися до потреб різних (і потенційно паралельних) сервісів, що працюють на потужних вузлах (у хмарі, на планшетах тощо). Деякі сервіси, наприклад, підписуються на сповіщення про температуру (давачі відповідають за повідомлення сервісу про досягнення певного порогового значення), а потім переналаштовують давачі на майже безперервне надсилання даних, щоб сервіс міг точно відслідковувати зміну температури.

Застосування моделей під час виконання (або будь-якої техніки динамічної адаптації) на виконавчих вузлах з обмеженими ресурсами (наприклад, обчислювальних вузлах на базі мікроконтролерів) є набагато складнішим з наступних причин:

1. Час простою: Мікроконтролери часто містять програмне забезпечення, яке безпосередньо керує фізичними пристроями. Перезавантаження або заморожування цих мікроконтролерів може мати серйозні наслідки, якщо мікроконтролери керують критично важливими для безпеки пристроями, або неприємні та помітні ефекти, якщо вони керують пристроями для комфорту.

2. Використання енергозалежної пам'яті (ОЗП): Динамічний розподіл пам'яті є наріжним каменем, який забезпечує динамічну адаптацію.

Мікроконтролери зазвичай мають лише кілька кілобайт оперативної пам'яті, і це обмеження не дозволяє зберігати в пам'яті кілька конфігурацій одночасно.

3. Використання постійної пам'яті: Постійна пам'ять необхідна для того, щоб процес адаптації мав транзакційні властивості, що дозволяє відновити стан мікроконтролера у випадку перезавантаження після збою. EEPROM є поширеним типом постійної пам'яті, вбудованої в мікроконтролери, зазвичай з дуже обмеженим розміром. Цей тип пам'яті також має обмежений термін служби з точки зору кількості операцій запису. Подібно до твердотільних дисків [72], запис до EEPROM слід розподіляти між комірками пам'яті, щоб оптимізувати термін служби всієї пам'яті.

4. Відновлення: Здатність до відновлення є критично важливою для вбудованих систем, які схильні до збоїв (наприклад, тимчасової втрати живлення). Мікроконтролери повинні перезавантажуватися і відновлювати свою останню конфігурацію досить швидко, щоб йти в ногу зі змінами конфігурації загальної архітектури.

КФС, що покладаються на великий набір автономних давачів, мають вартісні та енергетичні обмеження, що вимагає дешевих і енергоефективних платформ, здатних працювати протягом тривалого періоду часу з мінімальним обслуговуванням на місці (наприклад, заміною батарей). Це особливо актуально у сфері моніторингу навколишнього середовища: моніторинг морських розливів нафти, прогнозування повеней [73], моніторинг якості повітря або радіаційний моніторинг, з наступних причин:

1. Їх спрощена архітектура є надійною і передбачуваною: мікроконтролери можуть працювати в широкому діапазоні температур (зазвичай від -40 до 85 градусів Цельсія), вологості, живлення і мають фіксовану кількість циклів для виконання заданої операції.

2. Їхні енергетичні потреби (і тепловиділення) дуже низькі: 8-бітний мікроконтролер, що працює на частоті 32 кГц, зазвичай споживає менше 0,05 Вт (менше 0,5 Вт на частоті 1 МГц), не потребуючи жодного радіатора. Таким чином, вони можуть працювати дуже довго від батареї.

3. Їх спрощена архітектура дозволяє масове виробництво, що робить мікроконтролери дуже дешевими для розгортання навіть у великих кількостях.

У порівнянні з повноцінними обчислювальними вузлами, дешеві мікроконтролери страждають від накладних витрат на адаптацію, які випливають з їх апаратної технології з точки зору часу адаптації або зносу пам'яті: динамічне резервування типів компонентів вимагає написання програми у флеш-пам'яті. Тому реалізація концепцій Kevoree для вузлів мікроконтролерів залежить від точного компромісу між гнучкістю та типовими експлуатаційними витратами. Пошук легкого рішення для кожного з описаних вище рівнів реконфігурації є одним з головних викликів μ-kevoree.

3.3 Практичне дослідження

Було використано приклад «розумної» будівлі, щоб перевірити роботу Kevoree на наборі гетерогенних вузлів, включаючи, звичайно, деякі мікроконтролери. Різні системи (що покладаються на пристрої та протоколи) зазвичай розгортаються в будівлях для управління різними аспектами автоматизації будівлі, зокрема, комфортом (освітлення, кондиціонування повітря тощо), безпекою та захистом (виявлення диму та пожежі, спринклерів тощо). Ступінь гнучкості, яку пропонує система автоматизації будівель, часто дуже низька.

- Ці системи покладаються на фіксовану топологію каналів зв'язку. Давачі та виконавчі механізми часто повинні бути фізично з'єднані, що перешкоджає будь-якій майбутній реконфігурації або еволюції системи. Наприклад, давач руху вмикає всі світильники в коридорі. - Архітектура організована навколо центрального сервера. Коли пристрої фізично не з'єднані, вони зазвичай взаємодіють через центральний сервер, щоб виконувати правила, які керують системою на основі подій. Хоча оновлення цих правил можливе, для адаптації поведінки системи потрібен доступ до центрального сервера.

Мета Kevoree - безперешкодно розподілити бізнес-логіку та можливості динамічної адаптації на гетерогенних вузлах, починаючи від потужних серверів,

планшетів і закінчуючи простими пристроями, керованими мікроконтролерами. У повсякденному житті це дозволить користувачам конфігурувати та переконфігурувати свої офіси "на льоту" зі смартфона (наприклад, визначати освітленість відповідно до навколишнього освітлення, температури тощо), тоді як деякими іншими проблемами буде керувати центральний сервер (наприклад, вмикати камери вночі). У кризовій ситуації такий безперебійний розподіл дозволив би аварійним службам впоратися з виходом з ладу деяких вузлів. Наприклад пожежники все одно зможуть отримати доступ до даних, що надаються низькорівневими давачами, і обчислити значущу контекстну інформацію в системі прийняття тактичних рішень, незважаючи на втрату вузлів.

3.4 µ-Kevoree

У цьому підрозділі описано, як основні концепції Kevoree були перенесені на мікроконтролери. µ-Kevoree повністю узгоджено та сумісно з версіями Java та Android, що виходять з розробки.

Типи – у Kevoree типи компонентів і типи каналів інкапсулюють бізнес-логіку; вони генеруються як структури C. Надані порти типів компонентів зіставляються з методами, так що клієнтські компоненти (яким потрібні порти того ж типу) можуть викликати ці методи і, в кінцевому підсумку, передавати дані. Властивості Kevoree, які можна динамічно оновлювати, просто відображаються на локальні змінні, що містяться у цій структурі. Локальний планувальник запобігає одночасним викликам цих змінних. Необхідні порти створюються як локальні структури, які за бажанням можуть посилатися на зв'язаний екземпляр каналу. Аналогічно, типи каналів генеруються як звичайні структури на мові C. Прив'язки вихідних каналів генеруються як внутрішня структура масиву, що дозволяє динамічно розподіляти та зберігати посилання на зовнішні надані порти.

Асинхронна передача повідомлень – як і в реалізаціях Kevoree для вузлів Java та Android, µ-Kevoree зіставляє кожен порт та канал з актором [74]. Точніше, перед кожним захищеним методом генерується черга FIFO. Потім диспетчер

(локальний для кожного компонента) відповідає за відправлення повідомлень, що стоять у цих чергах, до потрібного методу. Цей локальний планувальник керується глобальним планувальником, описаним нижче.

Планувальник екземплярів – на кожному вузлі глобальний планувальник екземплярів відповідає за підтримання узгодженого стану свого вузла, застосовуючи наступну стратегію балансування:

- Періодичне виконання: глобальний планувальник періодично викликає локальний планувальник кожного екземпляра компонента, який оголосив періодичне виконання;

- Тригерне виконання: глобальний планувальник викликає локальний планувальник кожного компонента, який має непорожню чергу повідомлень.

Глобальний планувальник також періодично перевіряє наявність зовнішніх повідомлень, пов'язаних з динамічною адаптацією, як описано у наступних двох підрозділах.

3.5 Мікропрограма для підтримки основних змін

Перепрошивка мікропрограми мікроконтролера з подальшим перезавантаженням пристрою - це простий спосіб адаптації мікроконтролерного вузла шляхом повної заміни його реалізації. Цей метод адаптації є прийнятним у деяких специфічних і контрольованих умовах (початкове виробництво, технічне обслуговування на місці тощо), оскільки пристрій фізично підключений до більш потужного вузла за допомогою лінії зв'язку з широкою пропускною здатністю (наприклад, дротового каналу зв'язку). У цьому випадку перепрошивка пам'яті контролера є досить безпечною (за умови, що код прошивки є безпечним), а також досить швидкою: перепрошивка всієї пам'яті шляхом завантаження нової прошивки та подальше перезавантаження пристрою займає лише кілька секунд. Однак цей метод є проблематичним, коли пристрої розгорнуті віддалено. Перезавантаження прошивки по повітрю є небезпечною маніпуляцією: прошивки, як правило, є об'ємними даними (порівняно з іншими даними, які зазвичай передаються бездротовими каналами зв'язку), і ймовірність виникнення

помилок зв'язку є вищою; це вимагає просунутих протоколів для обробки помилок. На практиці такий підхід суттєво впливає на час, необхідний для встановлення нової прошивки.

Запропонований підхід обмежує перепрошивку випадками, коли потрібно розгорнути нові типи компонентів. У зв'язку з цим мікроконтролери на основі C не забезпечують такої ж гнучкості, як вузли Java/OSGi, щодо динамічного резервування та завантаження класів [75]. Це, як правило, потрібно для початкового розгортання системи, коли всі заплановані типи компонентів забезпечені, або для значних еволюцій системи (наприклад, для роботи з новим типом пристрою, не передбаченим до початкового розгортання). У всіх інших випадках, таких як, наприклад, реконфігурація екземплярів компонентів, наш підхід виконує часткове оновлення флеш-пам'яті.

3.6 Безшовна динамічна адаптація мікроконтролерів

Дотримуючись принципів model@runtime, наш процес динамічної адаптації повністю автоматизований, що позбавляє розробників від необхідності писати низькорівневі адаптаційні скрипти або переплутувати логіку адаптації з бізнес-логікою. Перед будь-якою адаптацією всі необхідні перевірки нової конфігурації виконуються на цільовій моделі. Оскільки мікроконтролерні вузли мають обмежену обчислювальну потужність, перевірки конфігурації виконуються на більш потужних вузлах на базі Java або Android. Ці перевірки спрямовані на виявлення невідповідності між запланованою конфігурацією та фізичними апаратними можливостями. Після цього етапу перевірки конфігурація використовується як вхідні дані для алгоритму генератора, який обчислює сценарій реконфігурації. Цей сценарій потім передається в компактному вигляді на залежні мікроконтролери. Оскільки в мережах бездротових давачів часто трапляються помилки зв'язку, ми уникаємо проблемних неузгодженостей станів мікроконтролерів, реалізувавши механізм відновлення на основі відкату.

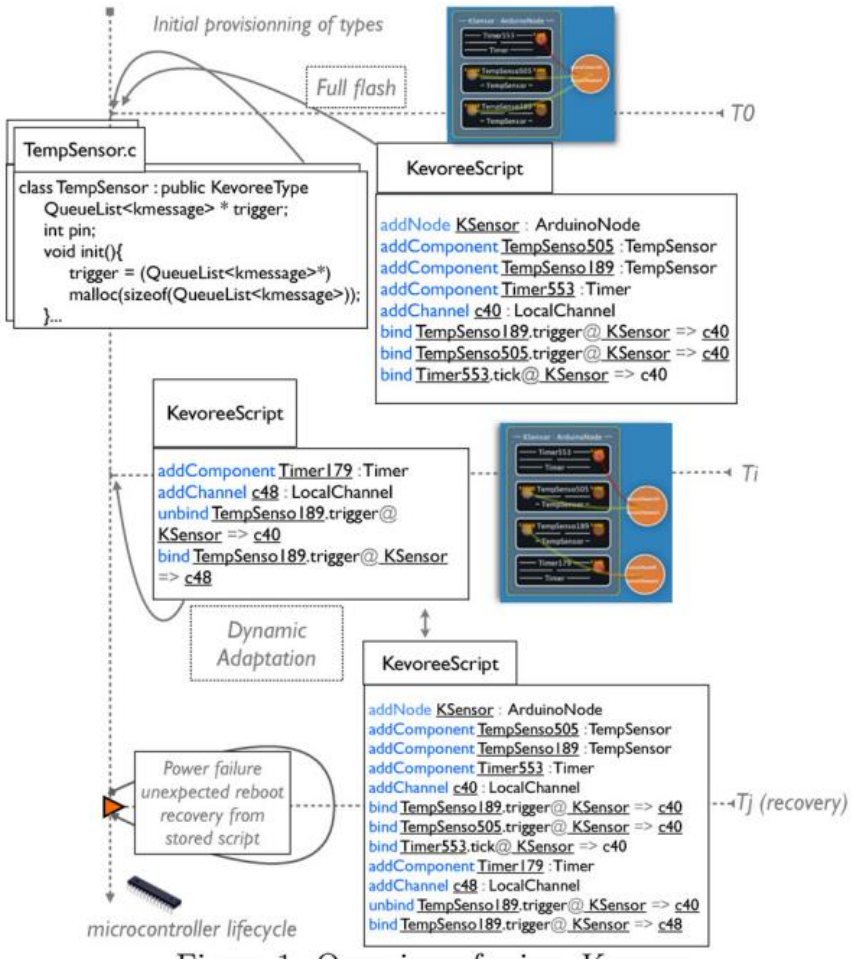

Рисунок 3.1 – Огляд мікро- Kevoree

У таблиці 3.1 перераховані рівні динамічної адаптації, що підтримуються різними технологіями вузлів Kevoree. Найпотужніші вузли здатні запускати програми на Java і реалізовувати всі рівні адаптації. Чим більш обмежені вузли, тим більше компромісів доводиться шукати.

Таблиця 3.1 – Рівні динамічної адаптації Kevoree

Динамічна адаптація	JavaSE	Android	Arduino
Параметричний	+	+	+
Архітектурний	+	+	+
Динамічне забезпечення	+	+	+/–
Управління за принципом «рівний рівному»	+	+/–	–

Незважаючи на значну обмеженість ресурсів, вузли Arduino все ще здатні підтримувати майже всі функції динамічної адаптації. У більшості випадків складні рішення будуть прийматися програмним забезпеченням, що працює на більш потужних вузлах. Прийняття рішень про адаптацію вимагає обробки правил адаптації, оптимізації цілей тощо, оскільки мікроконтролери зазвичай не мають достатньої обчислювальної потужності.

Відображення є фундаментальним принципом для досягнення динамічної адаптації. Але мова C не має підтримки примітивів, необхідних для побудови повноцінної моделі відображення; це перешкоджає прямому застосуванню техніки model@runtime на вузлах мікроконтролерів [76]. Було усунуто це обмеження, згенерувавши код для емуляції відбивного шару на цих мікроконтролерах. Під час генерації фреймворку ми припускаємо, що кількість типів є скінченною. Додавання або видалення типу означає регенерацію всього фреймворку. З цим припущенням про закритий світ типів, ми генеруємо плоский шар відображення, використовуючи вичерпне зіставлення шаблонів для екземплярів. Методи, які приймають екземпляри як параметр; ці методи надають наступні основні сервіси:

Алгоритм 3.1 – Основний сервіс μ-Kevoree

Function interruptScheduler(),resumeScheduler()
Function setProperty(instanceID,propID,propValue) : Bool
Function addBinding(channelID,componentID,portID) : Bool
Function removeBinding(channelID,componentID,portID) : Bool
Function createInstance(instanceID,typeID) : Bool
Function destroyInstance(instanceID) : Bool
Function exportCurrentState() : KevScript

Використано скрипт для експорту стану з метою оптимізації споживання пам'яті. Кожне текстове представлення, що використовується для експорту моделі відображення і пов'язане з визначеннями типів (наприклад, імена портів), локально кодується у статичній флеш-пам'яті для економії динамічної пам'яті.

Вбудований інтерпретатор KevScript використовує методи пласкої рефлексивності для реалізації базових сервісів (наприклад, життєвий цикл екземпляра, зв'язування та управління параметрами).

Отримавши сценарій Kevoree у стислому форматі, мікроконтролер виконує наступні завдання описані в алгоритмі 3.2:

Алгоритм 3.2 – Інерпретатор KevScript

```
Function interpretScript(script : KevScript)
  interruptScheduler()
  if permanentMemory.size >= PermanentMemoryLimit then
    resetPMemoryIndex()
    writeToPMemory(exportCurrentState())
  end if
  lastRecoveryPoint ← createRecoveryPointInPMemory()
  ∀ st, st ∈ script.statements → writeToPMemory(st)
  if executeFromPMemory(lastRecoveryPoint) then
    closeRecoveryPoint(lastRecoveryPoint)
  else
    rollback(lastRecoveryPoint)
  end if
  resumeScheduler()
```

Скрипти перевіряються незалежно від контексту зв'язку, а потім зберігаються. Нова конфігурація фіксується атомарним записом у пам'ять після її перевірки, таким чином реалізуючи механізм атомарних транзакцій. Ця техніка запобігає неповному збереженню пам'яті. У разі виникнення будь-яких проблем під час інтерпретації KevScript, відкат досягається простим перезавантаженням мікроконтролера. Рисунок 1 дає огляд цього процесу на прикладі реконфігурації давача температури. У час TO виштовхується повна конфігурація, що містить код на C та початковий KevScript. У момент часу Ti виштовхується KevScript, додаючи 2 екземпляри. У проміжку часу між Ti і Tj відбувається збій живлення, що призводить до відновлення з використанням збереженого в пам'яті KevScript.

3.7 Обчислювальний експеримент

Під час дослідження отримуються такі параметри:

Час простою: загальний час, необхідний мікроконтролеру для адаптації, включаючи завантаження нової конфігурації. Таким чином, ця метрика вимірює накладні витрати, спричинені мікроконтролером на управління власним станом відносно часу виконання доменних додатків.

Використання оперативної пам'яті (RAM): обсяг оперативної пам'яті, виділений для динамічного розподілу екземплярів компонентів, каналів та прив'язок. Таким чином, цей показник впливає на максимальну кількість екземплярів компонентів, каналів та зв'язувань, якими може керувати мікроконтролер.

Використання постійної пам'яті: обсяг постійної пам'яті, що використовується для зберігання сценаріїв реконфігурації та вплив стратегії зберігання на час життя пам'яті. Типи постійної пам'яті, такі як EEPROM, вбудована в 8-бітовий AVR, мають обмежену кількість циклів запису, сертифікованих для кожного байта, тим самим обмежуючи обсяг даних, що зберігаються.

Затримка відновлювального перезавантаження: час, необхідний мікроконтролеру для перезавантаження та відновлення останньої конфігурації після збою або втрати живлення.

Було використано реалістичні конфігурації, щоб оцінити наш підхід та оцінити накладні витрати, спричинені нашою динамічною платформою на основі компонентів для мікроконтролерів, порівняно зі статичними конфігураціями, які оновлюються шляхом перепрошивки всієї пам'яті. Всі експерименти були проведені з використанням реалізації вузла Kevoree Arduino, що працює на платі Arduino з мікроконтролером ATMEL AVR 328P. Цей процесор має 32 Кб флеш-пам'яті для зберігання програм, 2 Кб оперативної пам'яті та 1 Кб EEPROM. Для оцінки впливу типу пам'яті на результати було також використано флеш-пам'ять (microSD), підключену через шину SPI, як постійну пам'ять [77].

Наступні підрозділи показують наш конкретний експериментальний протокол і результати, в той час як в останньому підрозділі буде представлено приклад промислового використання для перевірки безперешкодної інтеграції пристроїв μKevoree в існуючу динамічну архітектуру.

3.7.1 Час простою

У цьому експерименті було налаштовано п'ять різних конфігурацій. Відповідні моделі Kevoree використовували різну кількість екземплярів для імітації змін між конфігурацією, що використовується вночі, та персоналізованою конфігурацією, що використовується вдень. Моделі були сконфігуровані з 4 вузлами, на кожному з яких було розміщено від 0 до 10 екземплярів. Екземпляри реалізовані на мові C, в середньому по 30 рядків коду кожен.

На першому кроці згенерувано мікропрограму нашого тестового мікроконтролера, яка містить всі визначення типів, що використовуються у цьому експерименті. Таким чином, цей крок включає генерацію коду, компіляцію та запис до флеш-пам'яті. Крім того, згенерований код автоматично перевіряється за допомогою давачів для вимірювання часу простою та використання пам'яті (EEPROM та SDRAM). Цей крок повторювався з затримкою в 100 мс з новою конфігурацією, яка обиралася випадковим чином; кожна нова конфігурація динамічно встановлювалася на заміну поточної конфігурації. Цей крок випадкової реконфігурації було повторено 500 разів. На рисунку 2 показано графіки вихідних даних, зібраних у цьому експерименті. Графік зверху показує, що використання оперативної пам'яті є постійним. Другий графік показує час простою на одну реконфігурацію, а третій і нижній графіки показують час простою і розмір сценарію відповідно.

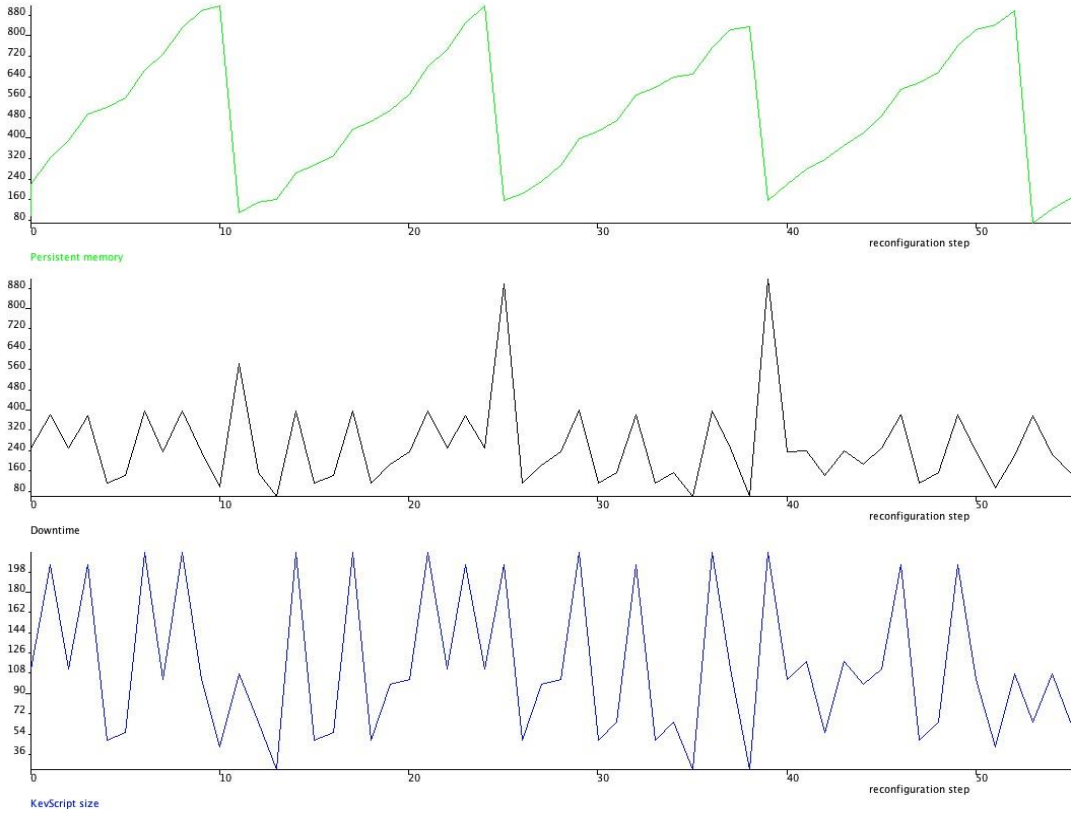

Рисунок 3.2 – Первинні результати експерименту

Результати експериментів та аналіз. Розгортання конфігурації шляхом перепрошивки всієї прошивки є дуже дорогим: час простою для розгортання початкової конфігурації становить 12,208 секунд. Таке високе значення пов'язане, головним чином, з тривалим часом передачі повної прошивки, а також з часом, необхідним для повного перезапуску мікроконтролера стандартним завантажувачем. Це значення варіюється в діапазоні +/-2 секунди.

Результати цього першого експерименту показують, що розмір скрипту реконфігурації сильно корелює з часом простою: коефіцієнт кореляції Спірмена між розміром скрипту та часом простою перевищує 0,9. Крім того, алгоритм стиснення, який використовується для зменшення розміру сценарію в EEPROM, також впливає на час простою. Очевидно, що виконання цього завдання прямо корелює з вищими значеннями часу простою.

Після 500 циклів реконфігурації виміряно наступні екстремальні та середні значення:

- мінімальний час простою 58 мс, що в 210 (тобто 12208 / 58) разів швидше, ніж статичне миготіння;

- максимальний час простою 916 мс, що в 14 (тобто 12208 / 916) разів швидше, ніж статичне миготіння;

- середній час простою 235 мс, що в 52 (12208 / 235) рази швидше, ніж статичне миготіння.

Для кращого аналізу цих даних використано графік розподілу за процентами для значень часу простою, як показано в таблиці 3.2.

Таблиця 3.2 – Графік розподілу значень часу простою

%	0	5	25	50	75	95	100
Простої (мс)	58	59	139	221	248	398	916

На графіку на Рисунку 3.3 чітко видно, що значення часу простою згруповані навколо 220 мс. 95% значень нижче 400 мс і 75% нижче 250 мс. І лише 5% значень перевищують 400 мс, що пояснюється кроком стиснення EEPROM. Стратегія лінивого стиснення дозволяє нам обмежити кількість піків і утримати максимальне значення близько 200 мс. Найвищі значення часу простою систематично пов'язані зі зменшенням розміру EEPROM (спричиненим процедурою стиснення). Ми спостерігали 32 стиснення EEPROM під час 500 реконфігурацій, тобто 6,4% реконфігурацій викликають стиснення так, що вони можуть бути повністю збережені в EEPROM. Максимальне значення цих 32 піків простою становить 916 мс, мінімальне - 218 мс, а середнє - 580,815 мс. Це середнє значення значно вище, ніж середнє значення всього набору з 500 реконфігурацій (234,682 мс).

Під час цього експерименту зонди також контролювали SDRAM. Ні витоків пам'яті, ні фрагментації пам'яті не відбулося. Хоча SDRAM є стабільною, необхідно перевірити, що накладні витрати, спричинені фреймворком, дійсно дозволяють динамічно створювати реалістично велику кількість екземплярів, щоб відповідати потребам конкретного випадку використання. Наступний експеримент був спрямований на оцінку продуктивності (з точки зору динамічного створення екземплярів) нашого тестового мікроконтролера.

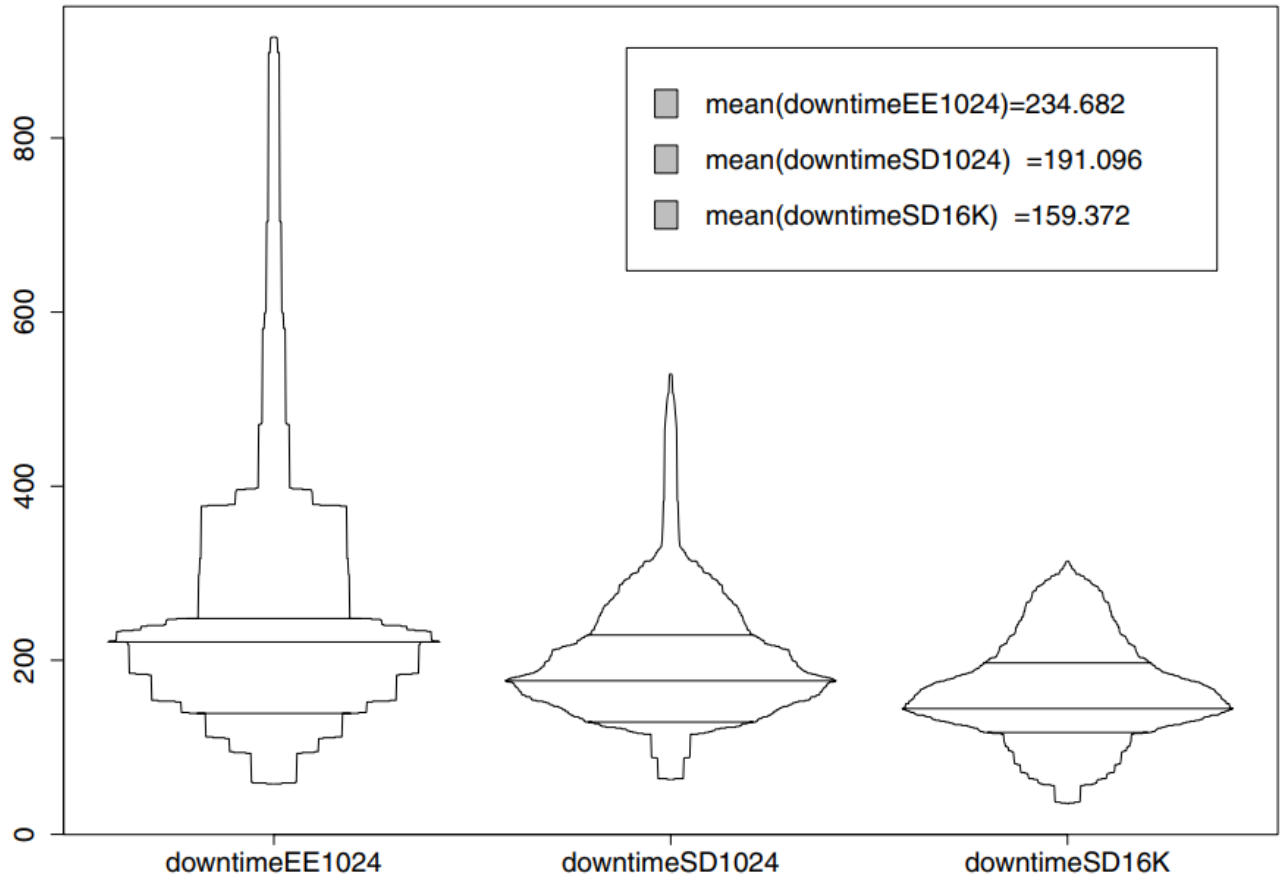

Рисунок 3.3 – Процентний розподіл часу простою флеш-пам'яті (в мс)

Результати експерименту та аналіз з іншим налаштуванням. Використано протокол з 500 ітерацій, але замінено EEPROM на зовнішню флеш-пам'ять (SD-карту) об'ємом 2 ГБ, підключену по шині SPI. Цей експеримент було повторено двічі: з об'ємом пам'яті 1кб (такий самий, як і в EEPROM) та 16кб; результати наведено в таблиці 3.3.

Таблиця 3.3 – Результати експерименту

Percentile(%)	0	5	25	50	75	95	100
ПростійSD 1K(ms)	63	88	129	176	229	324	529
ПростійSD 16K(ms)	35	56	117	145	197	297	314

Флеш-пам'ять має довший час ініціалізації, що пояснює те, що найнижчі значення для експерименту з флеш-пам'яттю вищі, ніж найнижчі значення в експерименті з EEPROM. Однак, швидкість запису флеш-пам'яті є високою, що призводить до гомогенізації часу простою, який більшу частину часу не

перевищує 200 мс. Можна помітити, що значне збільшення постійної пам'яті (16 кб) стримує піки часу простою і, відповідно, покращує середнє значення часу простою. Однак це не змінює суттєво розподіл основних значень.

3.7.2 Нестабільне використання пам'яті

Метою цього експерименту було точне визначення максимальної кількості екземплярів, які можуть поміститися в SDRAM. Було створено початкову конфігурацію з трьома об'єктами: таймером, перемикачем і каналом за замовчуванням. Кожні 100 мс конфігурацію розширювали, додаючи новий екземпляр комутатора. Для моніторингу SDRAM вводилися зонди.

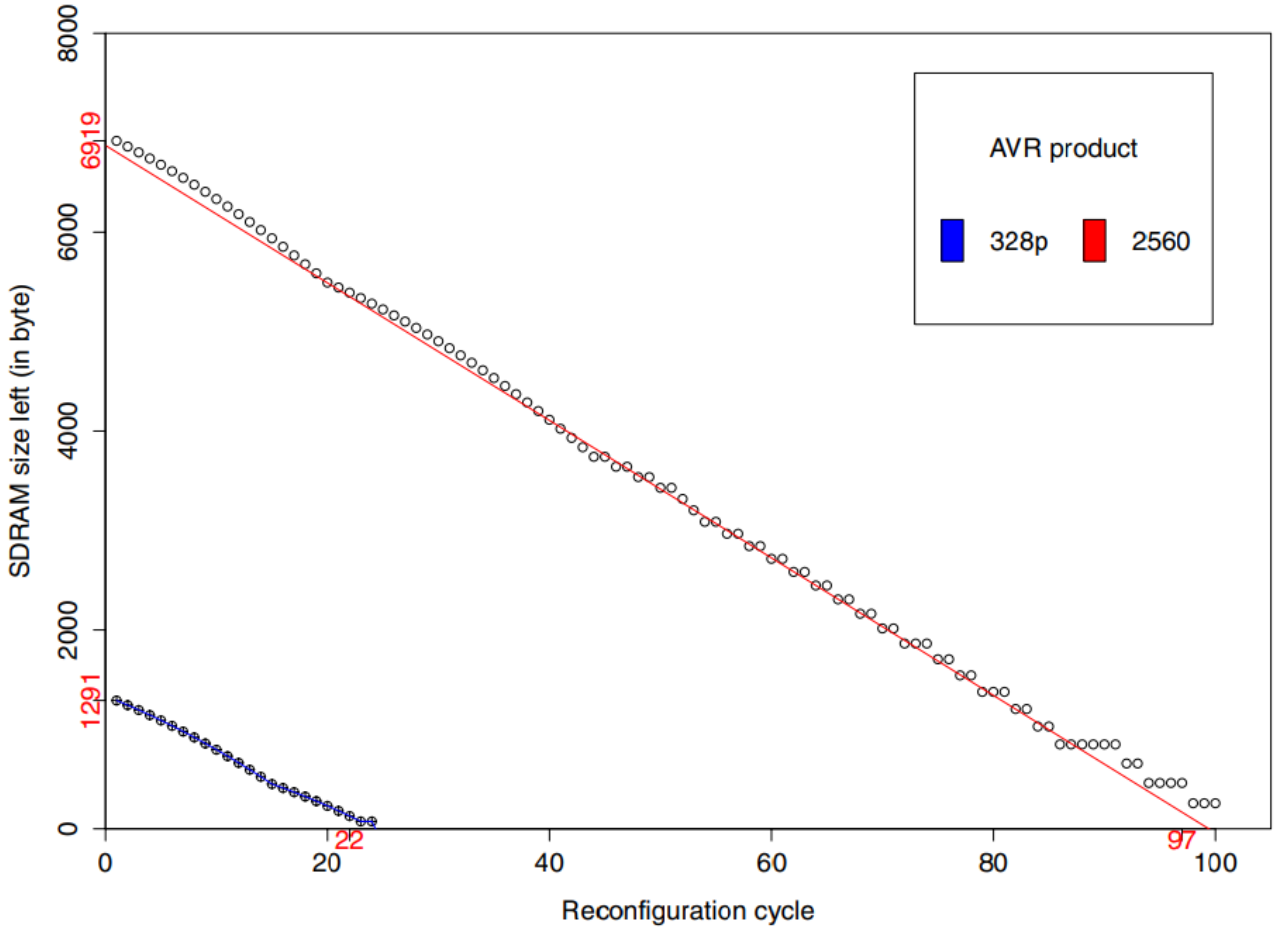

Рисунок 3.4 – Експеримент з ємністю SDRAM

На рисунку 3.4 показано, що пам'ять SDRAM заповнюється після 22 циклів, тобто наш тестовий мікроконтролер може обробляти 25 екземплярів (3

початкових плюс 22 додаткових). На практиці кількість пристроїв, якими керує один мікроконтролер, приблизно дорівнює кількості виводів, які вони мають. Крім того, мікроконтролери повинні мати можливість виконувати певний код для керування цими пристроями та виконувати певні обчислення. На тестовому мікроконтролері (22 виводи) Kevoree може керувати 25 екземплярами (як для обгортання фізичних пристроїв, так і для визначення певної оркестрації). Незважаючи на витрати пам'яті, наш підхід сумісний з поточними практиками.

Результати експериментів та аналіз з іншим налаштуванням. У цій установці використано ATMEL 2560 (4 КБ SDRAM) як тестовий мікроконтролер. AVR 2560 витримує навантаження в 100 екземплярів, тобто покращення на +300% екземплярів з +300% SDRAM. Як результат, кількість екземплярів більша, ніж кількість виводів (80) AVR 2560.

3.7.3 Постійне використання пам'яті

В результаті експериментів було виявлено, що 500/32 = 15.625 реконфігурацій можуть відбутися до того, як EEPROM (1 КБ) буде заповнений, що вимагає ущільнення з використанням нового початкового стану. Як і для твердотільних дисків [2], операції запису до EEPROM повинні бути розподілені по всій пам'яті, щоб розподілити знос. Алгоритм записує кожен байт перед обчисленням нового початкового стану. Кожен байт цієї пам'яті сертифіковано на 100 000 записів . Таким чином, якщо наприклад 100 реконфігурацій щодня (що набагато більше, ніж потрібно в більшості тематичних досліджень), кожен байт EEPROM буде записуватися в середньому 6,4 рази на день, забезпечуючи 15,625 днів (тобто близько 43 років) сертифікованого терміну служби для EEPROM.

В середньому, можна серіалізувати скрипти реконфігурації експерименту, використовуючи 16 байт. Оскільки алгоритм гарантує, що кожен байт записується до того, як пам'ять потрібно стиснути, ми можемо використати міркування попереднього параграфа для обчислення часу життя з більшим обсягом пам'яті.

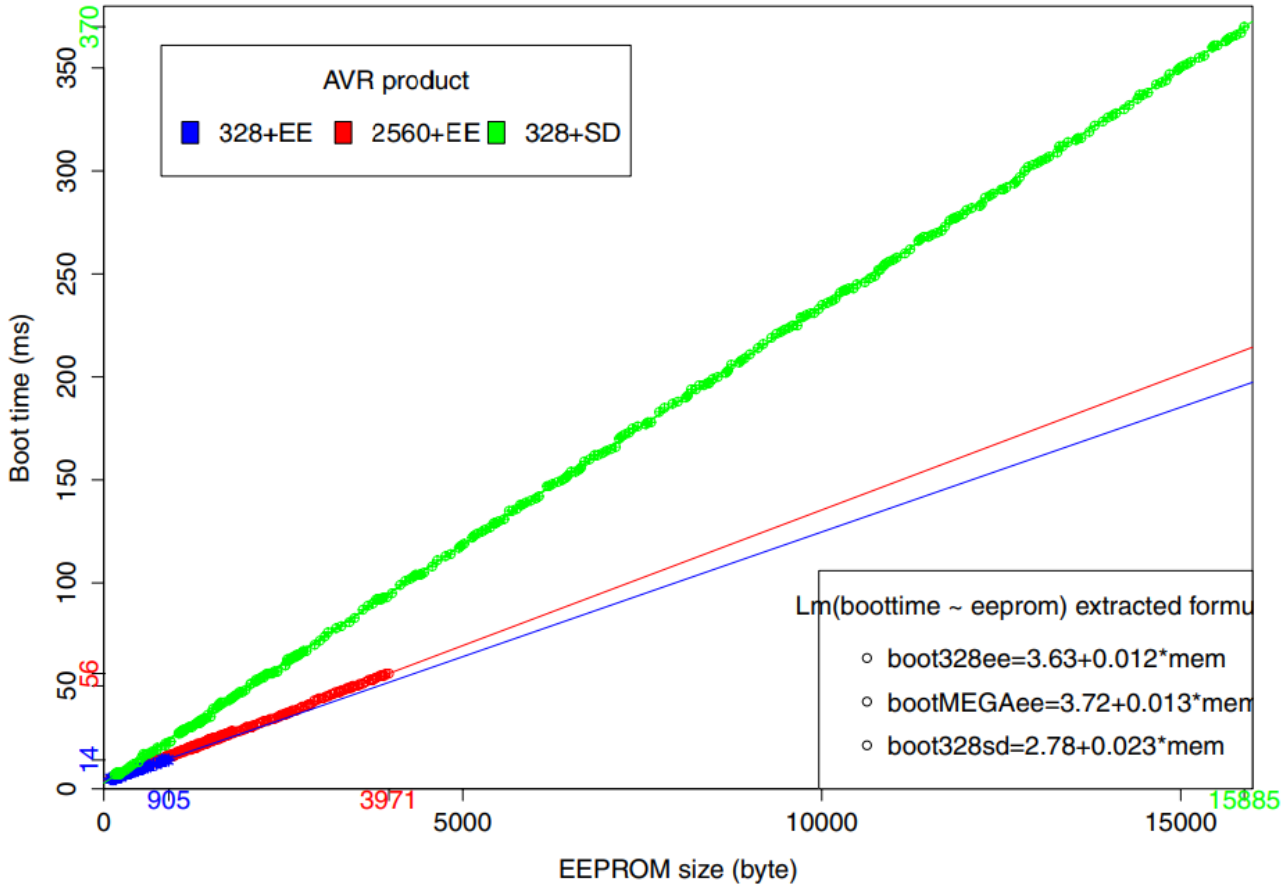

Рисунок 3.5 – Вплив постійного розміру пам'яті на час завантаження

3.7.4 Затримка перезавантаження при відновленні

У цьому експерименті використовувався набір конфігурацій з 50 циклами реконфігурації, які виконувалися кожні 2 секунди. Мікроконтролер фізично перезавантажувався між кожною реконфігурацією, і вставлявся новий зонд для вимірювання часу відновлення конфігурації після завантаження.

На рисунку 3.5 показано результати другого експерименту. Виявляється, що в найгіршому випадку з цим продуктом AVR (коли пам'ять EEPROM об'ємом 1 КБ майже заповнена) час завантаження становить приблизно 15 мс. У найкращому випадку (EEPROM майже порожній) час завантаження становить приблизно 3-4 мс. Це значення в основному пов'язане з низькою швидкістю зчитування EEPROM, вбудованої в AVR. Однак час відновлення конфігурації є прийнятним для більшості випадків використання, навіть у найгіршому випадку.

Отже, можна зробити висновок, що розмір сценарію в EEPROM має невеликий вплив на час завантаження з точки зору економії часу. Тому найкращою стратегією є використання всієї доступної пам'яті.

Було використано ту ж саму конфігурацію, але із заміною EEPROM на флеш-пам'ять (1 КБ і 16 КБ). Використання SD-пам'яті замість EEPROM призводить до збільшення часу завантаження. Було виявлено значення коефіцієнта 0,012 для EEPROM і 0,023 для SD. Це пов'язано з додатковими обчисленнями, необхідними для читання і запису SD-карти. На відміну від EEPROM, комунікаційна шина для доступу до флеш-пам'яті SD є зовнішньою по відношенню до мікроконтролера. Час ініціалізації, необхідний для конфігурації флеш-пам'яті, лінійно розподілений до межі 16 КБ. Понад цей розмір час ініціалізації стає більшим за 360 мс. Це значно перевищує час реконфігурації.

3.8 Результати експериментальних досліджень

Вибір типу та обсягу постійної пам'яті залежить від потреб конкретного випадку використання. У деяких випадках існує реальна потреба у відстежуваності, і історія системи повинна зберігатися, наприклад, для посмертного аналізу у випадку збою. В інших випадках більш важливими є показники адаптації та часу завантаження після збою. Розроблений в рамках цього підходу бенчмарк може бути корисним для емпіричного визначення того, який обсяг пам'яті слід використовувати.

Однак, використання зовнішньої пам'яті значно підвищує ціну такої платформи. Крім того, забезпечити атомарність реконфігурацій складніше через асинхронність протоколів передачі даних. Існуючі протоколи взаємодії з SD-картами (наприклад, MMC) не підходять для зберігання скриптів реконфігурації. Точніше, ці скрипти набагато коротші (близько 25 байт у наших експериментах), ніж мінімальний розмір кадру (512 байт), що вимагається цими протоколами, і це призводить до значних накладних витрат. На практиці більшість вбудованих пристроїв поєднують EEPROM і флеш-пам'ять. Kevoree дозволяє розробникам комбінувати різні типи пам'яті для різних цілей.

Накладні витрати μ-Kevoree. Наш фреймворк додає декілька джерел накладних витрат, особливо для управління динамічним створенням екземплярів компонентів та каналів. Для того, щоб кількісно оцінити накладні витрати, ми виміряли обсяги змінної та програмної пам'яті для програми HelloWorld, використовуючи звичайну мову C та прошивку Kevoree на мікроконтролері 328P AVR. У версії на звичайній мові C після послідовності завантаження залишалося 1842 вільних байти, тоді як у версії на Kevoree - 1604 байти. Це становить близько 11% від загального обсягу доступної оперативної пам'яті (242 байти з 2048). Звичайна версія на C використовувала 2.3 КБ пам'яті для прошивки, тоді як версія Kevoree використовувала 7.3 КБ, що дає накладні витрати близько 15% від загальних доступних 32 КБ. Вплив планувальника Kevoree на цикли обробки сильно залежить від програми, і ми наразі продовжуємо експериментувати, щоб обчислити ці накладні витрати.

Наш рівень синхронізації та зв'язку ввів накладні витрати менше 15% на обидві пам'яті, що є прийнятним значенням у випадку нашого IoT-додатку. Однак цей вплив слід оцінити більш глибоко для додатків, що працюють в режимі реального часу, приділяючи особливу увагу потребам компонентів з точки зору циклів обробки.

3.9 Висновок до третього розділу

У цьому розділі описано, як концепції Kevoree відображаються на вузли Arduino. Arduino - це програмно-апаратна платформа для створення прототипів електроніки з відкритим вихідним кодом, заснована на 8-розрядному мікроконтролері AVR. До плат Arduino можна підключати безліч давачів і приводів та програмувати на мовах сімейства C/C++. Хоча ми обрали Arduino для реалізації нашого підходу μ-Kevoree, його можна легко застосувати до інших сімейств мікроконтролерів (PIC, ARM тощо).

Реалізації прошивок часто кодуються вручну на C за допомогою методу спроб і помилок, з інтенсивною ручною та автоматизованою перевіркою на основі тестів. Більш досконалі методи (наприклад, методи MDE, запропоновані

ThingML [15]) спрямовані на створення статичного вихідного коду для мікроконтролерів. Такі методи можуть бути легко використані для генерації внутрішнього коду компонента, що не є сферою застосування Kevoree. Прошивка мікроконтролерів робить сильний акцент на використанні ресурсів (таких як пам'ять, процесор і енергоспоживання) і надійності: мікроконтролери можуть працювати протягом тривалого часу і відновлюватися в разі втрати живлення або зв'язку. Ми визнаємо, що ці властивості є критично важливими, і переваги, що надаються можливостями динамічної адаптації, не повинні ставити їх під загрозу. Наша робота спрямована на подолання статичної природи генерації коду зі збереженням усіх переваг низькорівневого дизайну коду.

Для цього запропонований підхід чітко розділяє структуру та поведінку: поведінка типів компонентів наразі реалізується вручну на мові C або на мові Wiring, використовуючи найсучасніші практики. Одним з основних внесків нашого підходу є визначення належної системи абстракцій для автоматизації управління логікою адаптації мікроконтролерів, шляхом виділення їх бізнес-логіки в окрему концепцію. Крім того, чітка структура компонентів з чітко визначеними входами і виходами також полегшує тестування на більш абстрактному рівні.

4 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

4.1 Питання щодо охорони праці

Тема кваліфікаційної роботи освітнього рівня «Магістр» присвячена динамічним мережам міської інфраструктури. Тому доцільно розглянути питання вдосконалення охорони праці в ІТ.

На перший погляд, робота за комп'ютером здається безпечною, але саме легковажність до неї може призвести до певних проблем у здоров'ї людини. Професія програміста та інших фахівців ІТ-технологій пов'язана з колосальним розумовим напруженням. Розробники – настільки захоплені люди, що навіть відволікаючись від роботи над проєктом, продовжують думати про роботу. Нерідко відпочинком вони вважають паралельну заміну основної діяльності, наприклад, читання профільної літератури, верстку сайтів, вивчення нових мов програмування. Однак мозок не може до безкінечності приймати виключно корисну інформацію, яку розробник прагне направляти в русло особистісного та професійного зростання.

Зараз багато ІТ-компаній обладнують свої офіси кімнатами відпочинку та лаунж-зонами, які забезпечують психофізіологічне розвантаження працівників. Адже окремим робочим столом з ноутбуком вже давно нікого не здивуєш. Тому, бажаючи підвищити продуктивність працівників, міжнародні компанії змагаються, перетворюючи нудні одноманітні офіси в креативні простори, де нові ідеї народжуються без титанічних зусиль.

При цьому не варто забувати, що умови праці програмістів також характеризуються можливістю впливу на них наступних небезпечних і шкідливих виробничих факторів: шуму; тепловиділень, причому шкоди організму можуть завдати не тільки високі, але і низькі температури; іонізуючих і неіонізуючих випромінювань: рентгенівське, інфрачервоне, електромагнітне випромінювання ВЧ і СВЧ діапазону; статичної електрики; недостатнє штучне та природнє освітлення; візуальні фактори: яскравість, контрастність, мерехтіння зображення, відблиски тощо.

За таких умов зростає роль та значення охорони праці, як системи правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних і лікувально-профілактичних заходів та засобів, спрямованих на збереження здоров'я і працездатності людини в процесі праці. Адже в кінцевому рахунку плоди науково-технічного прогресу можуть бути ефективними лише в тій мірі, в якій вони забезпечують людині безпеку, комфортність і зручність трудової діяльності.

Таким чином, використання новітніх інформаційно-комунікаційних технологій вимагає від фахівців ІТ-індустрії додержання певних правил та вимог з точки зору безпеки праці, її нормування з урахуванням віку працюючих та загального інформаційного навантаження, розробки та впровадження індивідуальних, щотижневих та щорічних режимів праці та відпочинку, які сприятимуть профілактиці перевтомлення і підвищенню розумової працездатності працюючих. Особливу роль в цьому напрямі повинні відігравати ергономічні заходи стосовно створення робочих місць, оптимізації взаємодії людини в рамках системи «оператор-термінал». Всі ці вимоги повинні бути втілені у відповідних нормативно-правових актах (стандартах підприємств), що регламентують різноманітні питання охорони та психології безпеки праці фахівців ІТ-індустрії/

Поява та впровадження нових інформаційно-комунікаційних технологій зумовлює необхідність подальшого вдосконалення охорони праці фахівців ІТ-індустрії.

З метою належного правового забезпечення необхідно розширити ти доповнити перелік основних професій комп’ютерної галузі у національному класифікаторі ДК-003-2010, а також підготувати відповідний випуск у кваліфікаційному довіднику посад фахівців ІТ-індустрії, що сприятиме вирішенню питань їх соціального захисту, пенсійного забезпечення, атестації робочих місць основних професій за умовами праці на предмет подальших певних видів пільг та компенсацій за важкі шкідливі і небезпечні умови праці.

Важливим напрямом стосовно визначення професійної придатності фахівців з інформаційних технологій є проведення психофізіологічної експертизи відповідно до 5 статті Закону України «Про охорону праці».

Робота з комп'ютерами нового покоління характеризується певним психофізіологічними перенавантаженнями, втомою зорового аналізатора, гіпокінезією, відсутність диференційованих норм праці при роботі з новою комп'ютерною технікою в залежності від віку, статі, категорії зорової роботи, режимів праці і відпочинку (протягом робочого дня, тижня, щорічного режиму відпусток).

Все це потребує розробки нових нормативно-правових актів з регламентації праці та відпочинку фахівців ІТ-індустрії і стандартів підприємств, центрів комп'ютерної техніки, центрів інформаційних технологій, сучасних комп'ютерних класів.

Особлива роль з точки зору збереження та відновлення здоров'я працюючих в комп'ютерні' галузі належить попереднім та періодичним наглядам з подальшої психофізіологічної експертизи і встановленням професійної придатності при роботі з комп'ютерами нового покоління, який супроводжується виникненням певних факторів професійного ризику електротравматизму при їх ремонті та обслуговуванні. В цьому зв'язку необхідне запровадження експертизи на предмет безпечної експлуатації ПЕОМ, тобто офіційне підтвердження фактичних параметрів електробезпеки, їх відповідності вимогам нормативної документації фахівців, які проводять таку експертизу повинні пройти навчання і перевірку знань відповідно до вимог ДНАОП 0.00-8.20-99. За результатами експертизи повинні прийматися рішення про відповідність ПЕОМ нормам безпеки, терміни чергової експертизи, оформлюються протоколи вимірювань і випробувань, проведені у разі потреби розрахунки та експертний висновок [78].

Для підвищення розумової працездатності то зорової роботи повинна здійснюватися ергономічна оптимізація в рамках системи «оператор-термінал», яка сприятиме результативній фізичній та інтелектуальний працездатності і відновленню психосоматичного здоров'я фахівців ІТ-індустрії.

4.2 Питання щодо безпеки в надзвичайних ситуаціях

Тема кваліфікаційної роботи освітнього рівня «Магістр» присвячена використанням кіберфізичних систем. Тому доцільно розглянути питання охорони праці користувачів ПК.

Широке розповсюдження отримали персональні комп'ютери. Однак їх використання загострило проблеми збереження власного та суспільного здоров'я, вимагає удосконалення існуючих та розробки нових підходів до організації робочих місць, проведення профілактичних заходів для запобігання розвитку негативних наслідків впливу ПК на здоров'я користувачів.

Заходи з охорони праці користувачів ПК необхідно розглядати в трьох основних аспектах: соціальному, психологічному та медичному.

У соціальному плані розв'язання цих проблем пов'язане з опти-мізацією умов життя, праці, відпочинку, харчування, побуту, розвитком культури, транспорту.

Значне місце у профілактиці розладів здоров'я належить психології праці. Тому заходи, пов'язані з формуванням раціональних виробничих колективів, у яких відсутня психологічна несумісність, сприяють зменшенню нервово-психічного перенапруження, підвищенню працездатності та ефективності праці.

Особливої значущості у користувачів відеодисплейних терміналів набуває психоемоційний стрес, який більшою або меншою мірою проявляється у кожного з них.

Оскільки цю проблему відразу вирішити неможливо, доцільно на рівні підприємства, організації послідовно усувати такі виробничі умови, які є сприятливими для розвитку емоційного стресу.

Значна роль у профілактиці захворювань користувачів ПК відводиться медицині. Існує перелік профілактичних заходів для користувачів ПК, що включає як складові первинної профілактики здоров'я (професійний відбір), так і вторинної, яка направлена на зниження ймовірності розвитку перевтоми та

перенапруження. Ці комплексні заходи спрямовані на відновлення функціонального стану зорового та опорно-рухового апарату.

Наказ Мінсоцполітики від 14.02.2018 р. №207 «Про затвердження Вимог щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями» (далі – Вимоги), зареєстрований в Міністерстві юстиції України 25 квітня 2018 року №508/31960, поширюється на всіх суб'єктів господарювання незалежно від форм власності, організаційно-правової форми і видів діяльності та встановлюють мінімальні вимоги безпеки та захисту здоров'я під час здійснення роботи, пов'язаної з використанням екранних пристроїв незалежно від їхнього типу та моделі.

Окрім того, відповідно до Закону України «Про охорону праці» роботодавець зобов'язаний:

- поінформувати працівників під розписку про умови праці та наявність на їх робочих місцях небезпечних та шкідливих виробничих факторів (фізичних, хімічних, біологічних, психофізіологічних), які виникають під час роботи з екранними пристроями та ще не усунуто, а також про можливі наслідки їх впливу на здоров'я працівників.
- забезпечити навчання і перевірку знань працівників з питань охорони праці та безпечного використання екранних пристроїв до початку роботи з ними, а також у випадках модифікації та організації роботи обладнання.
- роботодавець повинен вжити відповідних заходів, щоб забезпечити відповідність робочого місця працівника до цих Вимог.
- під час облаштування робочого місця працівника з екранними пристроями необхідно обирати таке устаткування, яке не створює зайвого шуму та не виділяє надлишкового тепла. Рівні шуму на робочих місцях осіб, які працюють з екранними пристроями, мають відповідати вимогам Санітарних норм виробничого шуму, ультразвуку та інфразвуку ДСН 3.3.6.037-99, затверджених постановою Головного державного санітарного лікаря України від 01.12.1999 р. №37.
- повинен за рахунок тривалості робочої зміни організувати внутрішні регламентовані перерви для відпочинку відповідно до Державних санітарних

правил і норм роботи з візуальними дисплейними терміналами електронно-обчислювальних машин ДСанПІН 3.3.2.007-98, затверджених постановою Головного державного санітарного лікаря України від 10.12.1998 р. №7 (далі – ДСанПІН 3.3.2.007-98).

- забезпечити за свій рахунок проведення медичних оглядів працівників відповідно до вимог Порядку проведення медичних оглядів працівників певних категорій, затвердженого наказом МОЗ від 21.05.2007 р. №246, зареєстрованого в Міністерстві юстиції України 23 липня 2007 року за №846/14113.
- роботодавець зобов'язаний за необхідності проводити лабораторні дослідження умов праці працівників з метою виявлення шкідливих і небезпечних факторів виробничого середовища, важкості та напруженості трудового процесу (зокрема щодо виявлення ризиків, пов'язаних із погіршенням зору, порушенням фізичного стану, стресом) та вживати заходів щодо усунення виявлених ризиків.

Окремо зазначаємо, що за характером трудової діяльності при роботі з відеотерміналами електронно-обчислювальних машин (далі – ЕОМ) та персональних електронно-обчислювальних машин (далі – ПЕОМ) виділено три професійні групи згідно з діючим класифікатором професій:

- розробники програм (інженери-програмісти) – виконують роботу переважно з відео терміналом (далі – ВДТ) та документацією при необхідності інтенсивного обміну інформацією з ЕОМ і високою частотою прийняття рішень. Робота характеризується інтенсивною розумовою творчою працею з підвищеним. напруженням зору, концентрацією уваги на фоні нервово-емоційного напруження, вимушеною робочою позою, загальною гіподинамією періодичним навантаженням на кисті верхніх кінцівок. Робота виконується в режимі діалогу з ЕОМ у вільному темпі з періодичним пошуком помилок в умовах дефіциту часу;
- оператори електронно-обчислюваних машин – виконують роботу, яка пов'язана з обліком інформації одержаної з ВДТ за попереднім запитом, або тієї, що надходить з нього, супроводжується перервами різної тривалості, пов'язана з виконанням іншої роботи і характеризується як робота з

напруженням зору, невеликими фізичними зусиллями нервовим напруженням середнього ступення та виконується у вільному темпі;

- оператор комп'ютерного набору – виконує одноманітні за характером роботи з документацією та клавіатурою і нечастими нетривалими переключеннями погляду на екран дисплея, з введенням даних з високою швидкістю, робота характеризується як фізична праця з підвищеним навантаженням на кисті верхніх кінцівок на фоні загальної гіподинамії з напруженням зору (фіксація зору переважно на документі), нервово-емоційним напруженням.

Відповідно до наведеної вище класифікації та згідно з вимогами ДСанПІН 3.3.2.007-98 є такі внутрішньозмінні режими праці та відпочинку при роботі з ЕОМ при 8-годинній денній робочій зміні в залежності від характеру праці:

- для розробників програм із застосуванням ЕОМ, слід призначати регламентовану перерву для відпочинку тривалістю 15 хвилин через кожну годину роботи.
- для операторів із застосуванням ЕОМ, слід призначати регламентовані перерви для відпочинку тривалістю 15 хвилин через кожні дві години роботи;
- для операторів комп'ютерного набору слід призначати регламентовані перерви для відпочинку тривалістю 10 хвилин після кожної години роботи.

При 12-годинній робочій зміні регламентовані перерви повинні встановлюватися в перші 8 годин роботи аналогічно перервам при 8-годинній робочій зміні, а протягом останніх 4-х годин роботи, незалежно від характеру трудової діяльності, через кожну годину тривалістю 15 хвилин.

Отже, встановлені відповідно до вимог ДСанПІН 3.3.2.007-98 внутрішньозмінні режими праці та відпочинку при роботі з електронно-обчислювальними машинами повинні входити в тривалість робочої зміни [78].

4.3 Висновок до четвертого розділу

В третьому розділі кваліфікаційної роботи описано охорону праці користувачів ПК та безпечне поводження з електронними пристроями.

Україна надалі вступає у еру широкого використання комп'ютерів у повсякденному житті, проте процес узгодження норм та умов роботи з цифровою технікою ще не завершено. У порівнянні з іноземними ІТ-компаніями, де цей процес вже укорінився, українські підприємства відстають, але підходять до завершальних етапів адаптації.

Не дивлячись на те, що робота за комп'ютером може здаватися безпечною, увага звертається на легковажність до цього процесу, яка може викликати проблеми у здоров'ї людини. Зокрема, висвітлено, що професія програміста та інших ІТ-фахівців супроводжується великим розумовим напруженням, а відсутність повного відпочинку може призвести до перевтоми та стресу.

У контексті охорони праці відзначено ряд факторів, які можуть впливати на здоров'я працівників ІТ-сфери, включаючи шум, тепловиділення, випромінювання, статичну електрику та освітлення. Підкреслено, що з урахуванням зростаючого значення цифрової техніки в організаціях, важливо розвивати та впроваджувати ефективні стратегії охорони праці для забезпечення безпеки та здоров'я працівників.

Важливу роль у вирішенні цих проблем відіграє психологія праці та медицина. Акцентується, що формування здорових виробничих колективів і психологічний комфорт є ключовими чинниками для зменшення нервово-психічного напруження та підвищення працездатності. Окрім того, наголошується на важливості медичних заходів та профілактичних програм для користувачів комп'ютерів з метою попередження ризиків та забезпечення стабільного функціонального стану зорового та опорно-рухового апарату.

У цілому, здійснено висновок, що із зростанням використання комп'ютерної техніки у різних сферах суспільства, необхідно вдосконалювати стратегії охорони праці, звертаючи увагу на соціальні, психологічні та медичні аспекти для забезпечення здоров'я та безпеки користувачів ПК.

ВИСНОВКИ

Основними рушійними силами розвитку та еволюції кіберфізичних систем (КФС) є скорочення витрат і часу на розробку, а також вдосконалення розроблених продуктів. Було оглянуто різні типи систем і пов'язаного з ними процесу переходу від мехатроніки до CPS і хмарних систем (IoT). Далі розглянуто вимогу, що методології CPS-дизайну повинні бути частиною міждисциплінарного процесу розробки, в рамках якого розробники повинні зосередитися не тільки на окремих фізичних і обчислювальних компонентах, але також на їх інтеграції та взаємодії. Тому виклики, пов'язані з CPS-дизайном, розглядаються в статті з точки зору фізичних процесів, обчислень та інтеграції відповідно.

Для вирішення задач з моделювання динамічних мереж міської інфраструктури з використанням кіберфізичних систем у кваліфікаційній роботі застосовано μ-Kevoree, що вирішує задачі динамічності та еластичності безпосередньо в пристрях з обмеженими ресурсами, базуючись на понятті models@runtime. Цей рівень моделювання, який мікроконтролери виставляють під час виконання, дозволяє ефективно і безпечно міркувати (іншими вузлами Kevoree: Java або Android) для адаптації вузлів на базі мікроконтролерів. Зокрема, ця стаття зосереджена на проблемах, що виникають при перенесенні можливостей Kevoree та models@runtime на малопотужні мікроконтролери, а також на необхідних компромісах через жорсткі ресурсні обмеження.

Kevoree було протестовано за допомогою бенчмарків, щоб оцінити її придатність до використання у реальних умовах. Незважаючи на витрати по відношенню до статичного (неадаптивного) коду, ці тести показали, що 75% транзакційних реконфігурацій можуть бути виконані менш ніж за 250 мс, що є прийнятним значенням у багатьох тематичних дослідженнях. Це швидше в 50 разів, ніж повний перезапис прошивки в пам'ять. Також ці бенчмарки показали, що час, необхідний для перезавантаження мікроконтролера і відновлення його попередньої конфігурації, є лінійною функцією від розміру скрипта. Наприклад, завантаження з використанням 1 кБ пам'яті EEPROM займає від 3 до 15 мс, в той

час як цей розмір пам'яті є достатньо великим, щоб зберігати сценарій 15 послідовних реконфігурацій, перш ніж виникне потреба в ущільненні. Нарешті, бенчмарки показали, що Kevoree дозволяє розгортати екземпляри програмних компонентів у кількості, що перевищує доступну кількість виводів на мікроконтролері. Таким чином, можна прив'язати програмний компонент до кожного фізичного пристрою, керованого мікроконтролером, і розгорнути додатковий компонент для координації цих компонентів.

В першому розділі кваліфікаційної роботи освітнього рівня «Магістр»:

– Розглянуто еволюцію кфберфізичних систем;

– Висвітлено Інтернет речей як елемент КФС;

– Проаналізовано основні парадигми кібефізичних систем.

В другому розділі кваліфікаційної роботи:

– Описано основні виклики при розробці та інтеграції КФС;

– Досліджено архітектуру та поведінкову парадигму для КФС;

– Подано порівняльний опис функціонального та архітектурного дизайну кіберфізичних систем.

В третьому розділі кваліфікаційної роботи:

– Розроблено концепцію Kevoree;

– Запропоновано динамічну компонентну модель Kevoree;

– Спроєктовано мікропрограму для підтримки основних змін;

– Протестовано безшовну динамічну адаптацію мікроконтролерів.

У розділі «Охорона праці та безпека в надзвичайних ситуаціях» проаналізовано охорону праці користувачів ПК та безпечне поводження з електронними пристроями.

ПЕРЕЛІК ДЖЕРЕЛ

1 Kitchin, Rob. "The Data Revolution: A critical analysis of big data, open data and data infrastructures." The Data Revolution (2021): 1-100.

2 Liagkou, Vasiliki, et al. "Challenges and opportunities in industry 4.0 for mechatronics, artificial intelligence and cybernetics." Electronics 10.16 (2021): 2001.

3 Zhou, Yuchen, et al. "Cyber-physical-social systems: A state-of-the-art survey, challenges and opportunities." IEEE Communications Surveys & Tutorials 22.1 (2019): 389-425.

4 Ren, Ju, et al. "A survey on end-edge-cloud orchestrated network computing paradigms: Transparent computing, mobile edge computing, fog computing, and cloudlet." ACM Computing Surveys (CSUR) 52.6 (2019): 1-36.

5 Schranz, Melanie, et al. "Swarm intelligence and cyber-physical systems: concepts, challenges and future trends." Swarm and Evolutionary Computation 60 (2021): 100762.

6 Rymarczyk, Jan. "Technologies, opportunities and challenges of the industrial revolution 4.0: theoretical considerations." Entrepreneurial business and economics review 8.1 (2020): 185-198.

7 Serpanos, Dimitrios, et al. "Embedded artificial intelligence: The ARTEMIS vision." Computer 53.11 (2020): 65-69.

8 Rodriguez-Rincon, Daniela, et al. "Study on the proposal evaluation system for the EU R&I framework programme. Final Report." (2022).

9 Pomante, Luigi, et al. "Design and management of image processing pipelines within CPS: 2 years of experience from the FitOptiVis ECSEL Project." 2020 23rd Euromicro Conference on Digital System Design (DSD). IEEE, 2020.

10 de Amorim Silva, Rafael, and Rosana T. Vaccare Braga. "Enhancing future classroom environments based on systems of systems and the internet of anything." IEEE Internet of Things Journal 7.10 (2020): 10475-10482.

11 Ande, Ruth, et al. "Internet of Things: Evolution and technologies from a security perspective." Sustainable Cities and Society 54 (2020): 101728.

12 Nakagawa, Elisa Yumi, et al. "Industry 4.0 reference architectures: State of the art and future trends." Computers & Industrial Engineering 156 (2021): 107241.

13 Thomason, Jane. "Big tech, big data and the new world of digital health." Global Health Journal 5.4 (2021): 165-168.

14 Mcharek, Mehdi, et al. "Collaboration and multidisciplinary design optimization for mechatronic systems." IECON 2019-45th Annual Conference of the IEEE Industrial Electronics Society. Vol. 1. IEEE, 2019.

15 Azzouzi, Elmehdi, et al. "A survey on systems engineering methodologies for large multi-energy cyber-physical systems." 2019 IEEE international systems conference (SysCon). IEEE, 2019.

16 Cervo, Andrea, et al. "Decentralized line equipment detection and production control by multi-agent technology." IECON 2019-45th Annual Conference of the IEEE Industrial Electronics Society. Vol. 1. IEEE, 2019.

17 Weyns, Danny, et al. "A research agenda for smarter cyber-physical systems." Journal of Integrated Design and Process Science 25.2 (2021): 27-47.

18 Shi, Yang, and Kunwu Zhang. "Advanced model predictive control framework for autonomous intelligent mechatronic systems: A tutorial overview and perspectives." Annual Reviews in Control 52 (2021): 170-196.

19 Bork, Dominik, Dimitris Karagiannis, and Benedikt Pittl. "A survey of modeling language specification techniques." Information Systems 87 (2020): 101425.

20 Leander, Björn. Dynamic Access Control for Industrial Systems. Diss. Malardalen University (Sweden), 2023.

21 21. Hinkelman, Kathryn, et al. "Modelica-based modeling and simulation of district cooling systems: A case study." Applied Energy 311 (2022): 118654.

22 Masoom, Alireza, et al. "Modelica-based simulation of electromagnetic transients using Dynaωo: Current status and perspectives." Electric Power Systems Research 197 (2021): 107340.

23 Abugabbara, Marwan, et al. "Modelica-based simulations of decentralised substations to support decarbonisation of district heating and cooling." Energy Reports 7 (2021): 465-472.

24 Barbieri, Giacomo, Cesare Fantuzzi, and Roberto Borsari. "A model-based design methodology for the development of mechatronic systems." Mechatronics 24.7 (2014): 833-843.

25 Lee, Edward Ashford, and Sanjit Arunkumar Seshia. Introduction to embedded systems: A cyber-physical systems approach. MIT press, 2016.

26 Sagi, Surya Vamsi Varma, and Leonard Petnga. "Ontological Modeling of Time and Time-Based Reasoning for Systems of Systems." Recent Trends and Advances in Model Based Systems Engineering. Cham: Springer International Publishing, 2022. 165-176.

27 Lee, Sangil, and Kwangyeol Ryu. "Development of the Architecture and Reconfiguration Methods for the Smart, Self-Reconfigurable Manufacturing System." Applied Sciences 12.10 (2022): 5172.

28 Li, Jichun, and Yi-Tung Chen. Computational partial differential equations using MATLAB®. Crc Press, 2019.

29 Schranz, M., and M. Sende. "Modeling swarm intelligence algorithms for cps swarms." ACM SIGAda Ada Letters 40.1 (2020): 64-73.

30 Diaz, Anna, et al. "Sustainable product development in a circular economy: Implications for products, actors, decision-making support and lifecycle information management." Sustainable Production and Consumption 26 (2021): 1031-1045.

31 Leng, Jiewu, et al. "Blockchain-empowered sustainable manufacturing and product lifecycle management in industry 4.0: A survey." Renewable and sustainable energy reviews 132 (2020): 110112.

32 Mcharek, Mehdi, et al. "Collaborative design process and product knowledge methodology for mechatronic systems." Computers in Industry 105 (2019): 213-228.

33 Zheng, Chen, et al. "Interface model-based configuration design of mechatronic systems for industrial manufacturing applications." Robotics and Computer-Integrated Manufacturing 59 (2019): 373-384.

34 Akundi, Aditya, and Viviana Lopez. "A review on application of model based systems engineering to manufacturing and production engineering systems." Procedia Computer Science 185 (2021): 101-108.

35 Sacks, Rafael, et al. "Toward artificially intelligent cloud-based building information modelling for collaborative multidisciplinary design." Advanced Engineering Informatics 53 (2022): 101711.

36 Liu, X. L., et al. "Industrial blockchain based framework for product lifecycle management in industry 4.0." Robotics and computer-integrated manufacturing 63 (2020): 101897.

37 Xu, Li Da. "The contribution of systems science to Industry 4.0." Systems Research and Behavioral Science 37.4 (2020): 618-631.

38 Andronie, Mihai, et al. "Big Data Management Algorithms, Deep Learning-Based Object Detection Technologies, and Geospatial Simulation and Sensor Fusion Tools in the Internet of Robotic Things." ISPRS International Journal of Geo-Information 12.2 (2023): 35.

39 Stark, Rainer, and Rainer Stark. "Major Technology 5: Product Data Management and Bill of Materials—PDM/BOM." Virtual Product Creation in Industry: The Difficult Transformation from IT Enabler Technology to Core Engineering Competence (2022): 223-272.

40 Stark, John. "Product lifecycle management (PLM)." Product Lifecycle Management (Volume 1) 21st Century Paradigm for Product Realisation. Cham: Springer International Publishing, 2022. 1-32.

41 Aranda-Mena, Guillermo, and Ron Wakefield. "Interoperability of building information—Myth of reality?." eWork and eBusiness in Architecture, Engineering and Construction. ECPPM 2006. CRC Press, 2020. 127-133.

42 Pal, Kamalendu. "Internet of things and blockchain technology in apparel manufacturing supply chain data management." Procedia Computer Science 170 (2020): 450-457.

43 Alam, Tanweer. "Cloud Computing and its role in the Information Technology." IAIC Transactions on Sustainable Digital Innovation (ITSDI) 1.2 (2020): 108-115.

44 Watanabe, Kentaro, Takashi Okuma, and Takeshi Takenaka. "Evolutionary design framework for Smart PSS: Service engineering approach." Advanced Engineering Informatics 45 (2020): 101119.

45 Fradi, Mouna, et al. "Conflict management for mechatronic systems design." 2021 IEEE International Symposium on Systems Engineering (ISSE). IEEE, 2021.

46 Lv, Zhihan, et al. "Trustworthiness in industrial IoT systems based on artificial intelligence." IEEE Transactions on Industrial Informatics 17.2 (2020): 1496-1504.

47 Costa Júnior, Ademir Almeida da. "A maturity model based on ISO/IEC/IEEE 42010: 2011 to identify technical debt in software architecture." (2020).

48 Lu, Ye, et al. "A Comprehensive Control Method for Tendon-Sheath System using Friction Model-Based Angle Estimation and Feedforward-Feedback Control in Time-Varying Configurations." IEEE Transactions on Industrial Electronics (2023).

49 Dan, Ning, Yuhao Yuan, and Jin Feng. "Fault-Tolerant Control of Speed Sensor for PMSM Based on Improved Maximum-Likelihood Voting." 2019 IEEE 8th joint international information technology and artificial intelligence conference (ITAIC). IEEE, 2019.

50 Hariri, Reihaneh H., Erik M. Fredericks, and Kate M. Bowers. "Uncertainty in big data analytics: survey, opportunities, and challenges." Journal of Big Data 6.1 (2019): 1-16.

51 Loquercio, Antonio, Mattia Segu, and Davide Scaramuzza. "A general framework for uncertainty estimation in deep learning." IEEE Robotics and Automation Letters 5.2 (2020): 3153-3160.

52 Tohidi, Seyed Shahabaldin, and Yildiray Yildiz. "Discrete adaptive control allocation." 2021 American Control Conference (ACC). IEEE, 2021.

53 Mittal, Anuj, Caroline C. Krejci, and Michael C. Dorneich. "An agent-based approach to designing residential renewable energy systems." Renewable and Sustainable Energy Reviews 112 (2019): 1008-1020.

54 Lan, Kai, and Yuan Yao. "Integrating life cycle assessment and agent-based modeling: a dynamic modeling framework for sustainable agricultural systems." Journal of Cleaner Production 238 (2019): 117853.

55 Dai, Erfu, et al. "Agent-based model of land system: Theory, application and modelling framework." Journal of Geographical Sciences 30 (2020): 1555-1570.

56 Bueno, Adauto, Moacir Godinho Filho, and Alejandro G. Frank. "Smart production planning and control in the Industry 4.0 context: A systematic literature review." Computers & Industrial Engineering 149 (2020): 106774.

57 Langmann, Reinhard, and Michael Stiller. "The PLC as a smart service in industry 4.0 production systems." Applied Sciences 9.18 (2019): 3815.

58 Schade, Florian, et al. "Dynamic Partial Reconfiguration for Adaptive Sensor Integration in Highly Flexible Manufacturing Systems." Procedia CIRP 107 (2022): 1311-1316.

59 Kovalenko, Ilya, Dawn Tilbury, and Kira Barton. "The model-based product agent: A control oriented architecture for intelligent products in multi-agent manufacturing systems." Control Engineering Practice 86 (2019): 105-117.

60 Karnouskos, Stamatis, et al. "Key directions for industrial agent based cyber-physical production systems." 2019 IEEE International Conference on Industrial Cyber Physical Systems (ICPS). IEEE, 2019.

61 Karnouskos, Stamatis, et al. "Industrial agents as a key enabler for realizing industrial cyber-physical systems: Multiagent systems entering industry 4.0." IEEE Industrial Electronics Magazine 14.3 (2020): 18-32.

62 Yang, Yiding, et al. "Learning dynamics via graph neural networks for human pose estimation and tracking." Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2021.

63 Lokey-Vega, Anissa, and Stephanee Stephens. "A batch of one: A conceptual framework for the personalized learning movement." Journal of Online Learning Research 5.3 (2019): 311-330.

64 Camilli, Matteo, Raffaela Mirandola, and Patrizia Scandurra. "Taming model uncertainty in self-adaptive systems using bayesian model averaging." Proceedings of the 17th Symposium on Software Engineering for Adaptive and Self-Managing Systems. 2022.

65 Moin, Armin, et al. "Supporting AI engineering on the IoT edge through model-driven TinyML." 2022 IEEE 46th Annual Computers, Software, and Applications Conference (COMPSAC). IEEE, 2022.

66 Song, Hui, et al. "Model-based fleet deployment in the IoT–edge–cloud continuum." Software and Systems Modeling 21.5 (2022): 1931-1956.

67 Abiodun, Oludare Isaac, et al. "Comprehensive review of artificial neural network applications to pattern recognition." IEEE access 7 (2019): 158820-158846.

68 Weyns, Danny. An introduction to self-adaptive systems: A contemporary software engineering perspective. John Wiley & Sons, 2020.

69 Longfei, Shangguan, and Priyantha Bodhi, eds. Mobile and Ubiquitous Systems: Computing, Networking and Services: 19th EAI International Conference, MobiQuitous 2022, Pittsburgh, PA, USA, November 14-17, 2022, Proceedings. Vol. 492. Springer Nature, 2023.

70 Ouareth, Selma, Soufiane Boulehouache, and Mazouzi Smaine. "Self-Adaptation Through Reinforcement Learning Using a Feature Model." International Journal of Organizational and Collective Intelligence (IJOCI) 12.4 (2022): 1-20.

71 Zhao, Kang, Zhiya Zuo, and Jennifer V. Blackhurst. "Modelling supply chain adaptation for disruptions: An empirically grounded complex adaptive systems approach." Journal of Operations Management 65.2 (2019): 190-212.

72 Stoica, Radu, et al. "Understanding the design trade-offs of hybrid flash controllers." 2019 IEEE 27th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS). IEEE, 2019.

73 Bencomo, Nelly, Sebastian Götz, and Hui Song. "Models@ run. time: a guided tour of the state of the art and research challenges." Software & Systems Modeling 18 (2019): 3049-3082.

74 Moin, Armin, et al. "Supporting AI engineering on the IoT edge through model-driven TinyML." 2022 IEEE 46th Annual Computers, Software, and Applications Conference (COMPSAC). IEEE, 2022.

75 Mamro, Oleksandr, and Andrii Lagun. "Micro-service support by module architecture application of the service platform for OSGI Java additions." Measuring equipment and metrology, 1 (81), 2020 1 (2020): 30-33.

76 Erazo-Garzón, Lenin, et al. "Models@ runtime and Internet of Things: A Systematic Literature Review." 2021 Second International Conference on Information Systems and Software Technologies (ICI2ST). IEEE, 2021.

77 Moin, Armin, et al. "A model-driven approach to machine learning and software modeling for the IoT: Generating full source code for smart Internet of Things (IoT) services and cyber-physical systems (CPS)." Software and Systems Modeling 21.3 (2022): 987-1014.

78 Стручок В.С. Безпека в надзвичайних ситуаціях. Методичний посібник для здобувачів освітнього ступеня «магістр» всіх спеціальностей денної та заочної (дистанційної) форм навчання / В.С.Стручок. — Тернопіль: ФОП Паляниця В. А., 2022. — 156 с

79 Vogel-Heuser, Birgit, Jay Lee, and Paulo Leitão. "Agents enabling cyber-physical production systems." at-Automatisierungstechnik 63.10 (2015): 777-789.

ДОДАТКИ

Додаток А

Тези конференцій

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ТЕРНОПІЛЬСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ПУЛЮЯ

М А Т Е Р І А Л И

XI НАУКОВО-ТЕХНІЧНОЇ КОНФЕРЕНЦІЇ

«ІНФОРМАЦІЙНІ МОДЕЛІ, СИСТЕМИ ТА ТЕХНОЛОГІЇ»

13-14 грудня 2023 року

ТЕРНОПІЛЬ
2023

УДК 004.732
Базан І.В., Коваль А.А.
(Тернопільський національний технічний університет імені Івана Пулюя, Україна)

ВИЯВЛЕННЯ КІБЕРАТАК В «РОЗУМНОМУ МІСТІ» НА ОСНОВІ МАШИННОГО НАВЧАННЯ

I. Bazan, A. Koval

DETECTING CYBERATTACKS IN A SMART CITY BASED ON MACHINE LEARNING

Технологічні винаходи змінили динаміку розвитку світу. Інфраструктура в кожній галузі автоматизується за допомогою Інтернету речей та бездротових мереж зв'язку. «Розумні міста» формуються на бездротовому зв'язку, де інфраструктура за своєю природою уможливлює велику кількість кібератак. Тому вразливості в «розумних містах» потребують належного вирішення. Сфера «розумних міст» досить різноманітна і має багато застосувань, включаючи електронний уряд, «розумні» будинки, інтелектуальний транспорт, телемедицину, «розумні» мережі, моніторинг, енергетику та багато іншого [1].

Безпека мереж передачі даних є темою для багатьох дослідників у всьому світі через постійне зростання кількості кібератак. Система виявлення вторгнень - це система, яка повинна ідентифікувати фальшиві пакети даних. Оптимальний алгоритм системи виявлення вторгнень балансує між високою точністю та показниками хибнонегативних і хибнопозитивних спрацьовувань. Крім того, основною її метою є виявлення можливих кібератак. «Розумні міста» потребують захищених каналів зв'язку, тому система виявлення вторгнень відіграє важливу роль [2]. Різноманітні технології, такі як ланцюги Маркова, машинне навчання, оптимізація та пуассонівський розподіл, використовуються для покращення систем виявлення сигнатур, аномалій та гібридних вторгнень.

Мережі IoT є вразливими до кібератак, тому в «розумному місті» протидія таким загрозам є великим викликом. DOS, DDOS, Sybil-атаки, SQL-ін'єкції та атаки зловмисного програмного забезпечення є поширеними типами атак в середовищі IoT, тому «розумні міста» постійно піддаються цим атакам. Результатом незахищеної мережі сенсорних вузлів може бути збій системи або припинення роботи сервісу, якщо не застосовувати превентивні засоби для забезпечення надійності і захищеності.

Існує багато рішень, які можуть бути використані відповідно до потреб захисту [3]. Найкращим підходом у виявленні різних загроз у «розумних містах» є машинне навчання. Машинне навчання для виявлення кіберзагроз у мережах «розумного міста». Існує три основні типи підходів машинного навчання: на основі аномалій, на основі сигнатур та гібридний. Виявлення на основі аномалій відбувається за допомогою інтелекту системи, навченого різними методами, підхід на основі сигнатур порівнює мережевий трафік з існуючими сигнатурами або шаблонами атак, що призводить до виявлення загроз, а гібридна система є сумішшю активів обох підходів, що робить її більш ефективною та точною, ніж обидва. Залежно від сценаріїв середовища, різні дослідники розробили різні типи системи виявлення вторгнень, використовуючи різні підходи, алгоритми з різними цільовими системами і порівнюючи точність і достовірність запропонованого ними алгоритму з іншими алгоритмами в своїх тематичних дослідженнях .

Література

1. Cimen, H., Palacios-García, E. J., et al. Smart-Building Applications: Deep Learning-Based, Real-Time Load Monitoring. IEEE Industrial Electronics Magazine, 15(2), 4-15.

2. Rincy N, T., & Gupta, R. Design and development of an efficient network intrusion detection system using machine learning techniques. Wireless Communications and Mobile Computing, 2021, 1-35.

3. Al-Turjman, Fadi, Hadi Zahmatkesh, and Ramiz Shahroze. "An overview of security and privacy in smart cities' IoT communications." Transactions on Emerging Telecommunications Technologies 33.3 (2022): e3677.

УДК 004.73:681.55
Коваль А.А., Базан І.В.
(Тернопільський національний технічний університет імені Івана Пулюя, Україна)

КІБЕРФІЗИЧНІ СИСТЕМИ: ВИКЛИКИ В СКЛАДНИХ МЕРЕЖАХ І РОЗПОДІЛЕНИХ СИСТЕМАХ

A. Koval, I. Bazan

CYBER-PHYSICAL SYSTEMS: CHALLENGES IN COMPLEX NETWORKS AND DISTRIBUTED SYSTEMS

Складні мережі є невід'ємною частиною повсякденного життя. Прикладами таких мереж є транспортні та телефонні мережі, інтернет, тощо. В останнє десятиліття аналіз і керування складними мережами, що складаються з декількох динамічних вузлів, привертає велику увагу в різних галузях. Зокрема, було закладено основи керування та синхронізації великих складних динамічних мереж з певними типами топології. У математиці складну динамічну мережу можна описати за допомогою графа. Кожен вузол такого графа представляє фундаментальний елемент з певною динамікою, а межі представляють інтерактивну топологію мережі. Розробка систем розпізнавання топології є важливим аспектом в складних мережах. Застосування таких систем можна знайти в різних галузях науки і техніки. Наприклад, якщо в мережі зв'язку, електромережі або Інтернеті трапляється значна несправність, дуже важливо визначити місце розташування дефектної ділянки, що стає можливим із застосуванням систем розпізнавання топології.

Задумані і реалізовані мережеві системи управління стикаються з багатьма проблемами, пов'язаними з обчислювальним часом, програмним забезпеченням, змінними часовими затримками, відмовами, реконфігурацією і розподіленими системами підтримки прийняття рішень. Розробка протоколу, що гарантує якість обслуговування в реальному часі в бездротових мережах, узгодження між дизайном закону управління і складністю реалізації в реальному часі, подолання різниці між системами безперервного і дискретного часу, а також надійність великомасштабних систем – виклики, які стоять в галузі кіберфізичних систем (КФС). Високі вимоги до надійності та безпеки гетерогенних компонентів, які взаємодіють у складному, комбінованому фізичному середовищі і працюють у кількох просторових і часових масштабах, вимагають застосування фреймворків, алгоритмів, методів і ресурсів. Математична модель, фізична модель, аналіз і алгоритм, дизайн системи і розподіл, з пов'язаними з ними непередбачуваністю і ризиком, є фундаментальними теоріями фізичного світу. Ключовим елементом КФС є давачі які є апаратними мостами між фізичним і кібернетичним світом, для яких необхідні методи обробки сигналів. У комплексному погляді КФС включає в себе стабільність, оптимізацію і управління розподіленими і цифровими системами.

Розподілені кіберфізичні алгоритми, які можуть бути використані в CPS для виявлення та аналізу подій, надають кожному вузлу повну інформацію системи навіть при зміні топології. CPS охоплюватиме різні аспекти соціального та економічного життя, матиме широкий вплив і стане орієнтиром для комп'ютерних наук та інших галузей. Очікується, що при проектуванні та виробництві майбутніх інженерних систем з новими технологіями, які значно перевищують сучасні стандарти автономності, функціональності, доступності, надійності та захисту даних, кіберфізичні системи матимуть важливе значення.

Література

1. Xia, Y., et al (2019). Introduction to focus issue: Complex network approaches to cyber-physical systems. Chaos: An Interdisciplinary Journal of Nonlinear Science, 29(9).

2. Mishra, A., et al (2023). Emerging technologies and design aspects of next generation CPS with a smart city application perspective. IJSAEM, 14(Suppl 3), 699-721.

3. Canizo, M., Conde, A., Charramendieta, S., Minon, R., Cid-Fuentes, R. G., & Onieva, E. (2019). Implementation of a large-scale platform for cyber-physical system real-time monitoring. IEEE Access, 7, 52455-52466.