Vol. 2, No. 1, 2020

https://doi.org/10.23939/csn2020.01.102

УДК 004.9

СТВОРЕННЯ АПАРАТНО-ПРОГРАМНОЇ ПЛАТФОРМИ ДЛЯ СУЧАСНОГО ЗАСТОСУВАННЯ ІНТЕРНЕТУ РЕЧЕЙ НА ОСНОВІ ТУМАННИХ ОБЧИСЛЕНЬ ІЗ ВИКОРИСТАННЯМ ТЕХНОЛОГІЙ CLOUD-NATIVE

А. Г. Шевченко, С. В. Пузирьов

Чорноморський національний університет імені Петра Могили,
кафедра комп'ютерної інженерії

Питання цифрової трансформації на цей момент є дуже актуальним у зв'язку з епідеміологічною ситуацією і переходом систем до цифрового середовища. IoT є одним із головних рушіїв цифрової трансформації. Internet of things (IoT) – це розширення всесвітньої мережі, яка об'єднує сенсори, контролери та інші різноманітні пристрої, так звані "things", які обмінюються даними між собою за допомогою всесвітньої мережі. Розроблення апаратно-програмного комплексу для організації туманних та граничних обчислень було поділено на три рівні: апаратний, оркестровий, прикладний, який поділяється на програмну та архітектурну частини. Апаратну частину реалізовано із застосуванням двох версій міні-комп'ютера Raspberry Pi: Raspberry Pi 4 та Raspberry Pi Zero, які підключаються в режимі master-slave. З боку оркестрової частини було використано технології K3S, Knative та Nuclio. Для реалізації програмної частини прикладного рівня було використано такі технології, як сервісна сітка Linkerd, система обміну повідомленнями NATS, реалізація протоколу RPC GRPC, бази даних TDengine, Apache Ignite, Badger. Архітектурну частину створено як стандарт розробки API, тому й можна застосовувати до різноманітних IoT програмних рішень будь-якою мовою програмування. Створену систему можна використовувати як платформу для побудови сучасних IoT-рішень за принципом туманних граничних обчислень.

Ключові слова: Internet of Things, IoT-платформа, Контейнерні технології, Digital Twin, API.

Вступ

Сучасний світ розвивається дуже стрімко, переходячи до цифрового середовища [1]. Це зумовлено певним набором переваг, що надає нам цифрове середовище порівняно з фізичними аналогами:

1. Уніфікований інтерфейс взаємодії (система менеджменту).
2. Автоматизація процесів.
3. Просте масштабування та розширення систем.
4. Адаптивність систем.

Як приклад можна навести сучасні електронні платіжні системи, такі як PayPal, Google Checkout, Payoneer та інші. Ці системи є прикладом цифрового трансформування фінансової системи. Вони дозволяють легко виконувати фінансові транзакції у цифровому середовищі, не використовуючи значні грошові суми фізично, незалежно від місцезнаходження відправника або отримувача.

Питання цифрової трансформації сьогодні є дуже актуальним у зв'язку з епідеміологічною ситуацією, адже все більше систем переходить у цифрове середовище [2]. Ще одним прикладом може бути так званий "digital workspace" – цифрова робоча область. Ця концепція полягає в створенні цифрової робочої області, до якої має можливість підключитися працівник з будь-якої точки, за наявності мережі Інтернет та відповідного електронного терміналу (телефон, комп'ютер, ноутбук тощо), який дозволяє розпочати свою роботу в мережі з мінімальними зусиллями із конфігурування [3].

Системи за рівнем цифрової трансформації можна поділити так:

1. Фізичні системи із частковою комп'ютеризацію або без неї. Тобто майже повна відсутність зв'язку фізичної системи з цифровим середовищем.

2. Цифрові системи без зв'язку з фізичними системами. Дані реального світу або не надходять взагалі, або надходять до системи через ручне введення. Також такі системи не можуть впливати на фізичне середовище, або їх вплив є незначним.

3. Комбіновані системи – системи, де фізичне та цифрове середовища повністю поєднано. Тобто цифрове середовище може автоматизовано збирати інформацію про фізичне та впливати на нього.

У попередні роки перші два типи систем були більш розповсюдженими ніж комбіновані системи, через певні проблеми, а саме:

- безпека та розподілений доступ;
- швидкодія;
- підключення до фізичного середовища (відсутність необхідних апаратних рішень).

Але сьогодні через зростання потужностей сучасних обчислювальних систем, здешевлення вартості пам'яті та появи великої кількості платформ збирання телеметрії зумовило поширення комбінованих систем [4].

До комбінованих систем належать IoT-системи, які здобули велику популярність. IoT-системи – це системи, які поєднують різні обчислювальні пристрої, сенсори, датчики, електронні пристрої в одну адаптивну обчислювальну мережу. Ринок IoT сьогодні становить 248 млрд. доларів США [5]. Це зумовлено тим, що IoT рішення допомагають покращити продуктивність та ефективність різних систем, покращити або автоматизувати технічні процеси та аспекти життя людей, тим самим збільшити прибуток сучасних IT-компаній [6].

Сьогодні існує велика кількість IoT-платформ – як апаратних, так і програмних, а також комбінованих. Головними проблемами сучасних IoT-платформ є:

1. Пропрієтарність або закритість систем.
2. Велика ціна.
3. Vendor-lock з боку програмних рішень.
4. Складність супроводження.
5. У деяких випадках низька адаптивність.

У цій роботі розглянуто апаратно-програмну платформу для побудови сучасних IoT-рішень, яка б використовувала:

- сучасні підходи – такі, як граничні та туманні обчислення;
- сучасні deploy-технології – такі, як контейнери, оркестратори, брокери повідомлень;
- сучасні архітектурні підходи: event-driven, microservices та serverless.

Ця платформа має вирішувати вищеописані проблеми, мати невелику ціну та бути побудованою на “відкритих” рішеннях.

Отже, метою роботи є побудова апаратно-програмної IoT-платформи на базі граничних та туманних обчислень із використанням cloud-native технологій, а також її тестування на прикладі системи розумного будинку. Ця платформа повинна відповідати сучасним вимогам до IoT-платформ [7–9], а саме:

- бути захищеною від несанкціонованого доступу;
- мати високу швидкодію;
- бути стандартизованою та мати чітко визначену архітектуру;
- бути відкритою до горизонтального та вертикального масштабування [7];
- вміти аналізувати оперативні дані у реальному часі, асинхронно аналізувати статистичні дані [9];
- надавати змогу керувати підключеними пристроями;
- вміти обробляти дані на «грані», тобто без використання хмарних обчислень в деяких випадках.

Технології Інтернету речей

Internet of things (IoT) можна по-різному визначати. Однак переважно це розширення всесвітньої мережі, яка об’єднує сенсори, контролери та інші різноманітні пристрої – так звані “things”, які обмінюються даними між собою за допомогою всесвітньої мережі. Одиницями Internet of things у цьому випадку можуть бути як “розумні” лампочки, розетки, двері, вентилятори, телефони, годинники, так і цілі “інтелектуальні” системи, наприклад, системи розумного будинку, підприємства, міста, одягу, автомобіля тощо [10].

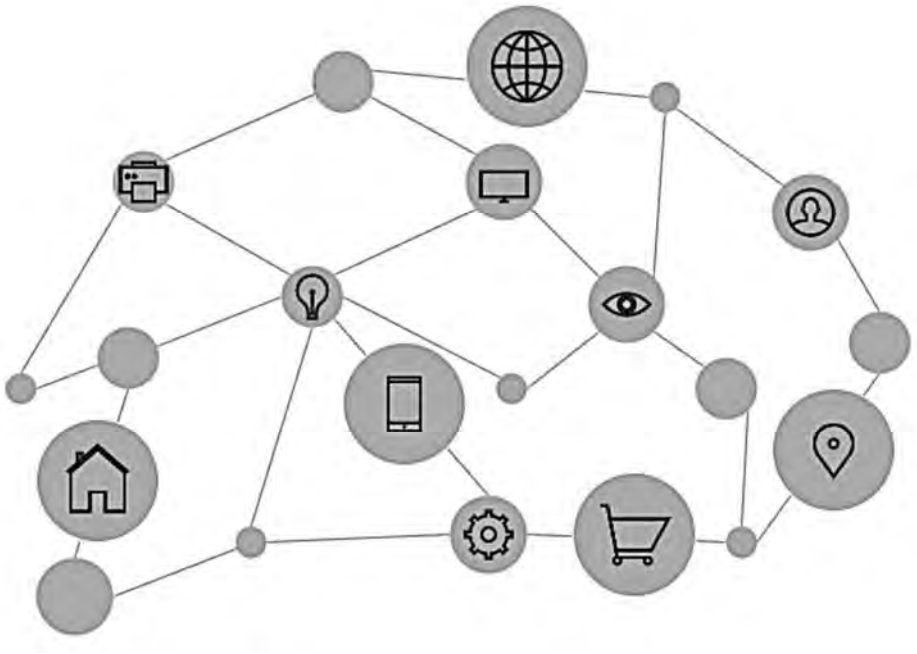

Рис. 1. Представлення концепції Інтернету речей (IoT) за допомогою графів

IoT-платформа – це повністю готова до експлуатації, монтажу та використання апаратно-програмна IoT система, яку можна гнучко налаштовувати під конкретні задачі. Реалізується як комплекс апаратної частини та готовий набор бібліотек, тобто API для взаємодії з системою на програмному рівні.

Граничні та туманні обчислення – це частина розподілених обчислень, яка містить обробку інформації та генерації контенту у безпосередній близькості від місця її збирання, виникнення або споживання [11]. Інколи ці типи обчислень описують як обчислення, які виконуються у певній системі, наближеній у просторі до систем збирання інформації до виконання хмарних обчислень як частина усього циклу обчислень [12].

Рис. 2. Багаторівнева діаграма представлення концепцій граничних та туманних обчислень

Туманні обчислення відрізняються від граничних ступенем наближення до точки збирання інформації у мережевій топології. Туманні обчислення знаходяться за дві або більше обчислювальних одиниць від кінцевої точки збирання інформації. Хоча загалом ці терміни взаємозамінні, останнім часом починає набувати популярність саме термін “граничні обчислення” [13, 14].

Контейнерні технології – це технології, які запускаються на обчислювальних машинах у вигляді контейнерів – ізольованих один від одного процесів-«пісочниць», та управляються певним оркестратором. Вони прискорюють розгортання серверних архітектур та створюють певний абстрактний рівень над апаратною частиною розподілених обчислень [15]. Для стандартизації контейнерних технологій було створено The Cloud Native Computing Foundation (CNCF) [16]. Cloud-native технології – це такі контейнерні технології, які можна легко розгорнути у певному контейнерному середовищі під управлінням певного оркестратора, а також технології, які підтримують, спрощують, автоматизують управління контейнерним середовищем. Головним критерієм визначення Cloud-native технологій є входження до CNCF офіційних проектів[17, 18].

Апаратно-програмний комплекс

У цій роботі розроблення апаратно-програмного комплексу для організації туманних та граничних обчислень поділено на три рівні та чотири частини:

- Апаратний рівень.
- Оркестровий рівень.
- Програмна частина (Прикладний рівень).
- Архітектурна частина (Прикладний рівень).

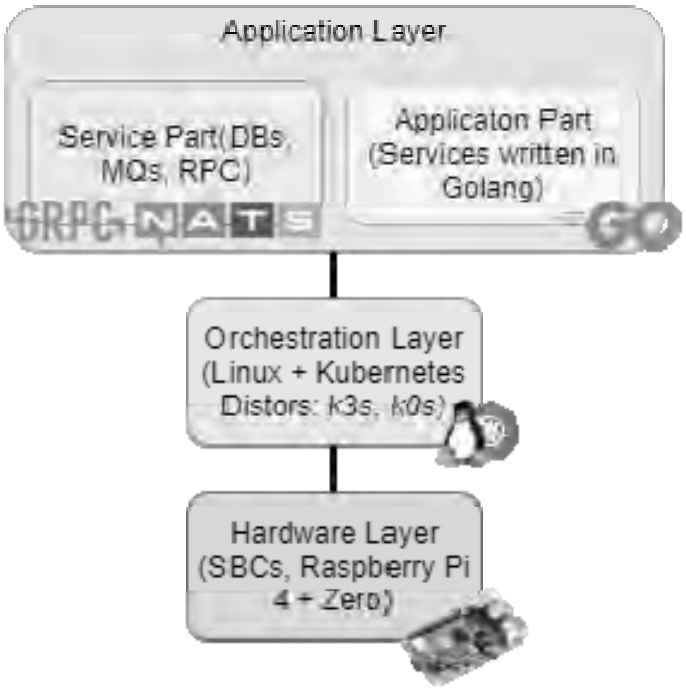

Рис. 3. Діаграма багаторівневої архітектури апаратно-програмного комплексу

Апаратний рівень

З боку апаратної частини можна використовувати будь-які мінікомп'ютери з підтримкою Linux систем [19]. У цьому випадку використано дві версії мінікомп'ютера Raspberry Pi: Raspberry Pi 4 (рис. 4) та Raspberry Pi Zero (рис. 5) через їх невелику ціну та популярність [20].

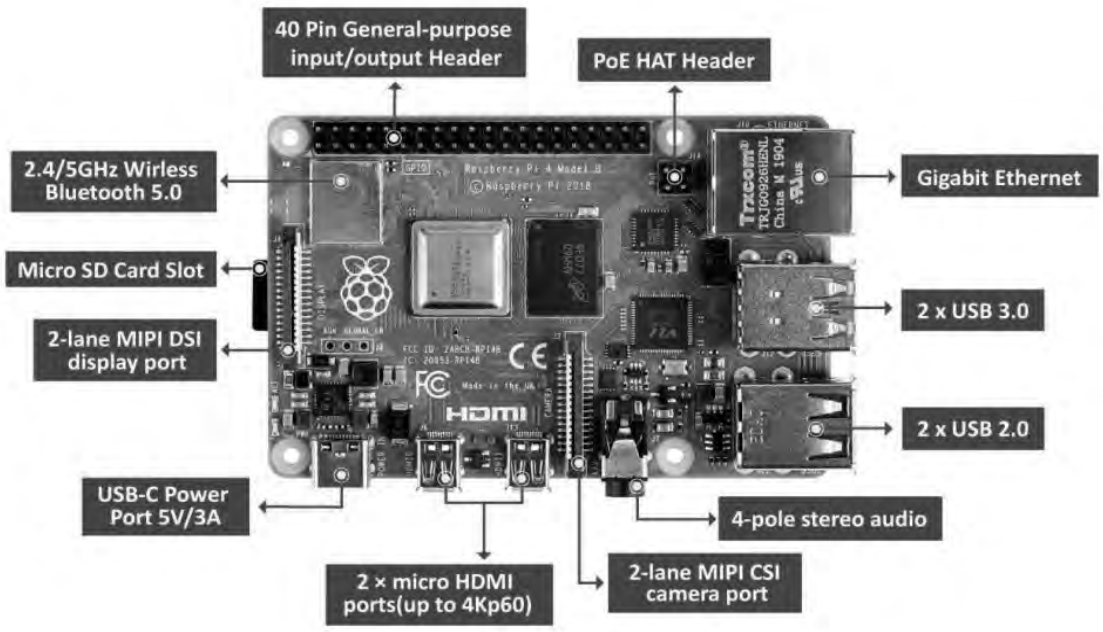

Рис. 4. Одноплатний мінікомп'ютер Raspberry Pi 4

У табл. 1 наведено деякі характеристики цього мікрокомп'ютера.

Таблиця 1

Характеристики Raspberry Pi

Processor	Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5 GHz
Memory	2 GB, 4 GB or 8 GB LPDDR4-3200 SDRAM (depending on model)
Wireless	2.4 GHz and 5.0 GHz IEEE 802.11ac wireless, Bluetooth 5.0, BLE
Ports	Gigabit Ethernet, 2 USB 3.0 ports; 2 USB 2.0 ports, Raspberry Pi standard 40 pin GPIO header, 2 × micro-HDMI ports, 2-lane MIPI DSI display port, 2-lane MIPI CSI camera port
Processor	Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5 GHz
Memory	2 GB, 4 GB or 8 GB LPDDR4-3200 SDRAM (depending on model)

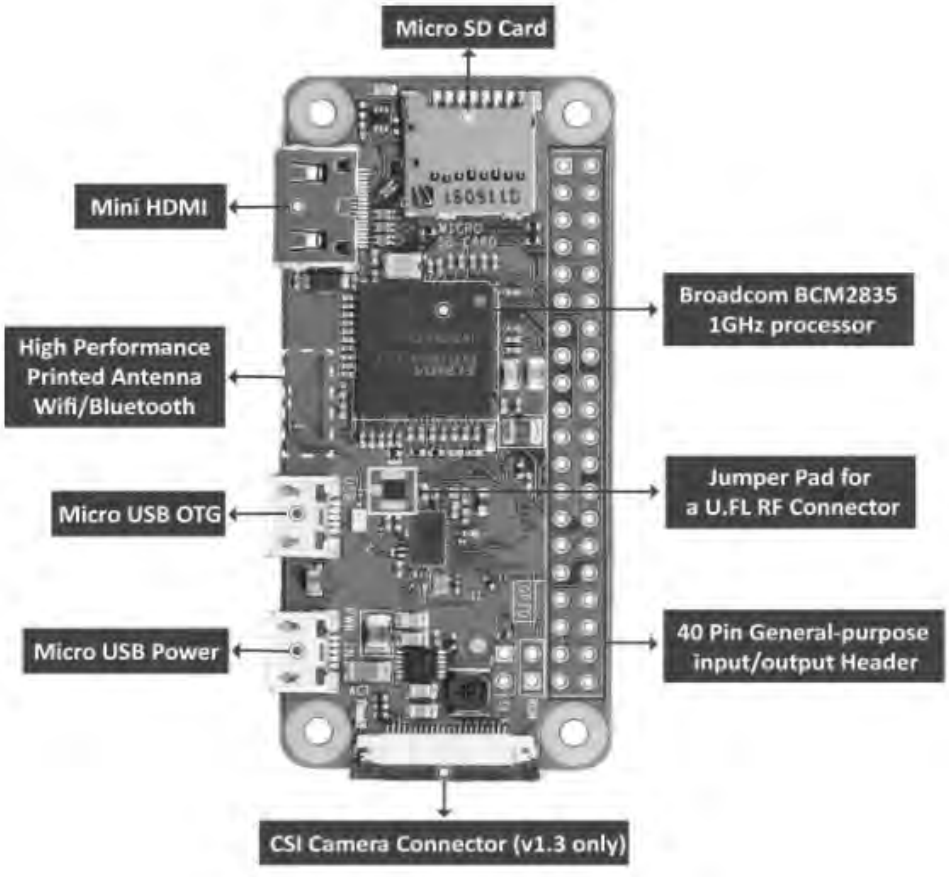

Рис. 5. Одноплатний мінікомп'ютер Raspberry Pi Zero

У табл. 2 наведено деякі характеристики мікрокомп’ютера Raspberry Pi Zero.

Таблиця 2

Характеристики Raspberry Pi Zero

Processor	1 GHz, single-core CPU
Memory	512 MB RAM
Wireless	802.11 b/g/n wireless, LAN Bluetooth 4.1, Bluetooth Low Energy (BLE)
Ports	Mini HDMI and USB On-The-Go ports, Micro USB power, HAT-compatible 40-pin header, Composite video and reset headers, CSI camera connector

Ці мінікомп’ютери підключаються в режимі master-slave, тобто за топологією “зірка”, де Raspberry Pi 4-master, центральний керуючий комп’ютер, а декілька одиниць Raspberry Pi Zero-slaves, обчислювальні комп’ютери, які підключаються до Raspberry Pi 4. Підключення відбувається за допомогою Wi-Fi з’єднання, оскільки Raspberry Pi 4 має лише один Ethernet порт [21]. Надалі для розширення системи можна об’єднувати такі обчислювальні кластери в більші системи за допомогою різних мережевих топологій, з’єднуючи між собою керуючі (master) комп’ютери. Ця топологія копіює топологію оркестрової частини, тому була реалізована саме в такому вигляді. Діаграму апаратного рівня платформи зображено на рис. 6.

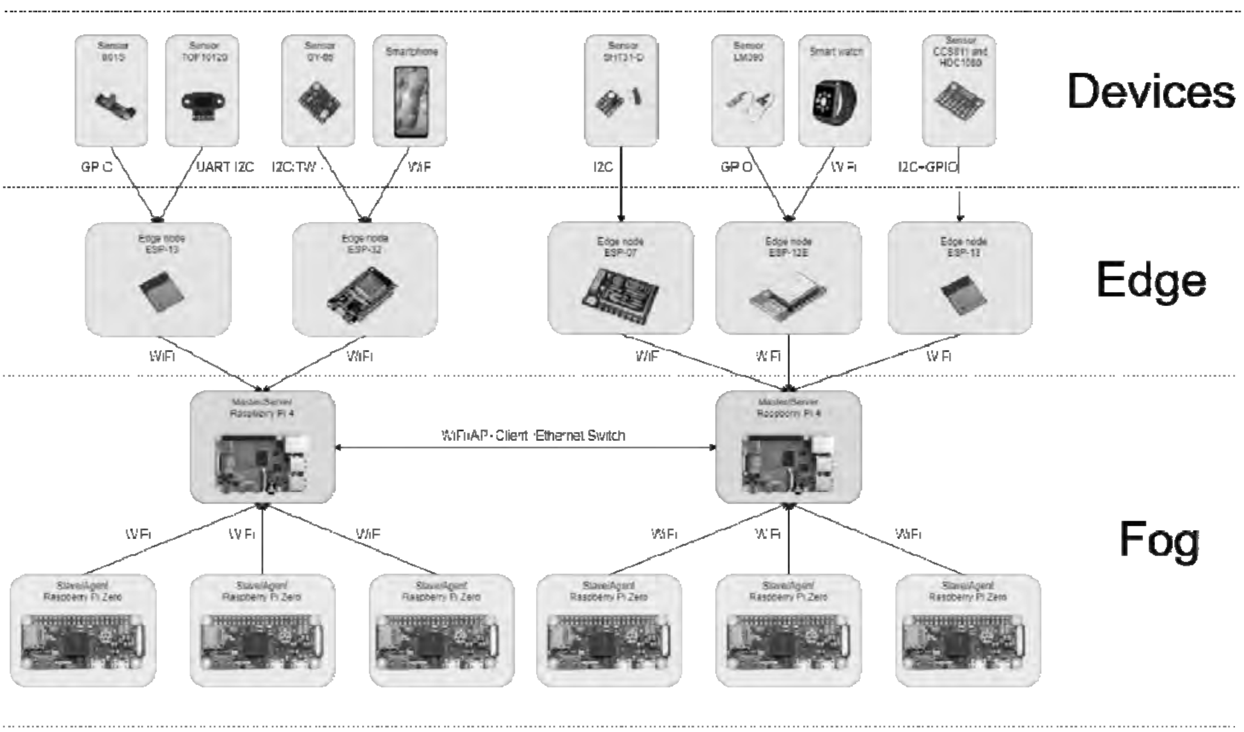

Рис. 6. Діаграма багаторівневої архітектури апаратного рівня платформи

Оркестровий рівень

З боку оркестрової частини було використано технології K3S, Knative та Nuclio.

K3S – це сертифікований Kubernetes дистрибутив, оптимізований для граничних обчислень, використання на мінікомп'ютерах [22]. Використовується в цьому комплексі як операційна система зі встановленим та налаштованим контейнерним середовищем.

Knative – це платформа-як-сервіс (PaaS, Platform as a Service), яка працює над Kubernetes, полегшує розроблення та розгортання, дозволяє динамічно масштабувати контейнери [23]. Knative використовують як основу для розгортання мікросервісної архітектури.

Nuclio – це платформа безсерверних обчислень, тобто обчислень, побудованих на архітектурі невеликих функцій та подій. Nuclio має високу швидкодію (виклик близько 400000 функцій за секунду), тому її було використано для реалізації безсерверної частини прикладної архітектури [24].

На рис. 7 представлено діаграму оркестрового рівня платформи.

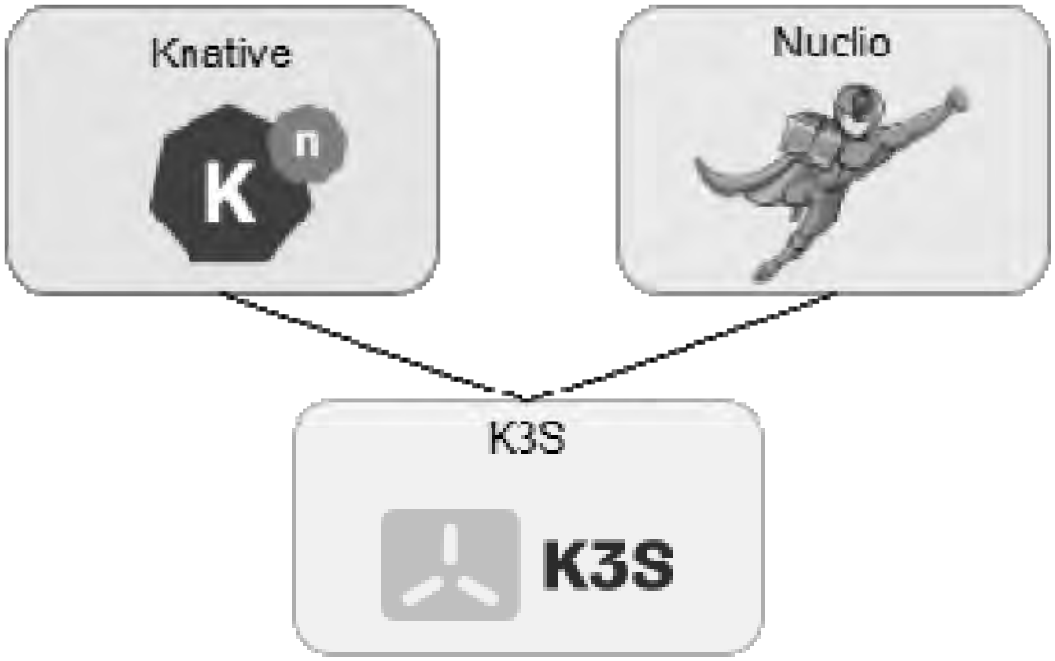

Рис. 7. Діаграма архітектури оркестрового рівня платформи

Програмна частина (прикладний рівень)

Для реалізації програмної частини прикладного рівня використано такі технології, як сервісна сітка Linkerd, система обміну повідомленнями NATS, реалізація протоколу RPC GRPC, бази даних TDengine, Apache Ignite, Badger.

Linkerd – це надлегка сервісна сітка (service mesh), створена для полегшення налаштування балансування (L7) та комунікації між мікросервісами у контейнерном середовищі [25].

NATS – це проста, безпечна та високопродуктивна система обміну повідомленнями з відкритим кодом [26]. NATS використовується на прикладному рівні як реалізацію асинхронного зв'язку між сервісами.

GRPC – це сучасна високопродуктивна система RPC із відкритим кодом, яка може працювати в будь-якому середовищі [27]. GRPC використовують на прикладному рівні як реалізацію синхронного зв'язку між сервісами для швидкого реагування на події в системі.

Apache Ignite – це горизонтально масштабована, відмовостійка розподілена обчислювальна in-memory база даних для побудови додатків у режимі реального часу [28]. Apache Ignite у системі використовують для збирання локальних даних сенсорів та як частину «hot path» аналітики.

TDengine – це платформа «великих даних» із відкритим кодом, розроблена та оптимізована для Інтернету речей (IoT), підключених транспортних засобів та промислових IoT рішень [29]. TDengine використовують у системі як постійне сховище даних та як частину «cold path\advanced» аналітики.

Badger є вбудованою, постійною та швидкою базою даних, яка працює за принципом "ключ-значення" (KV, key-value storage), написаною мовою програмування Go [30]. У проєкті використано як базу для реєстрації сенсорів та їх цифрових двійників.

Архітектурна частина (Прикладний рівень)

Архітектурну частину створено як стандарт розробки API, тому її застосовують до різноманітних IoT програмних рішень будь-якою мовою програмування, яка підтримує описані нижче парадигми. Задля ефективності, швидкості, роботи в реальному часі в цій апаратно-програмній платформі на прикладному рівні використано сучасні архітектурні рішення, такі як DDD, Microservices, Event-Driven, Serverless та концепція Digital Twin (Цифровий двійник).

Domain-driven design (DDD) – концепція проєктування програмних систем, яка полягає у розбитті програмних компонентів на моделі, відповідні до предметної області [31].

Event-Driven – архітектурний шаблон, який полягає у тому, щоб ділити сутності у програмному забезпеченні на генератори та споживачів подій, а також представляти усю модель спілкування через концепцію подій [32].

Microservices – архітектурний стиль, спосіб розпаралелювання програм представленням програмного забезпечення у вигляді невеликих сервісів, які спілкуються між собою — мікро-сервісів [33].

Serverless – архітектурний стиль, спосіб розпаралелювання програм, який полягає в розбитті програмного забезпечення на дуже дрібні частини – функції, які можуть запускатися паралельно та незалежно один від одного [34].

Концепція Digital Twin полягає у тому, щоб дзеркально відобразити у цифровому просторі системи та об'єкти реального світу за допомогою збирання телеметрії в реальному часі [35].

Усю архітектурну частину системи розбито на сервіси, мікросервіси та безсерверні функції. В основу покладено Event-Driven та багаторівневу архітектуру. Нижче перелічено основні архі-тектурні сутності розроблюваної системи.

Orchestration system – сервіс\сукупність мікросервісів автоматизованого балансування та управління IoT системою, відповідає за цілісну роботу усієї системи.

Management system – це сервіс\сукупність мікросервісів ручного управління та налаштування IoT системи, а також відображення метрик системи.

Device manager – сервіс\сукупність мікросервісів, який відповідає за реєстрацію нових пристроїв, створення та управління пристроями та їх цифровими двійниками.

Digital twin manager – сервіс\сукупність мікросервісів, який відповідає за творення та управління цифровими двійниками, які не мають фізичного аналогу в реальному світі, тобто відображають складні реальні системи чи процеси.

Digital twin – сервіс, який є цифровим двійником фізичної системи або агрегує інші цифрові двійники для створення цифрового двійника складнішої системи.

Operator – сервіс\сукупність мікросервісів, створений для виконання певних операцій над цифровими двійниками, таких як моніторинг, аналіз, обробка тощо.

Pipeline – це сукупність безсерверних функцій, призначених для послідовної або паралельної обробки даних, які надходять в або з цифрових двійників.

Як тестовий стенд розроблено архітектуру системи розумного будинку на основі цього стандарту (рис. 8).

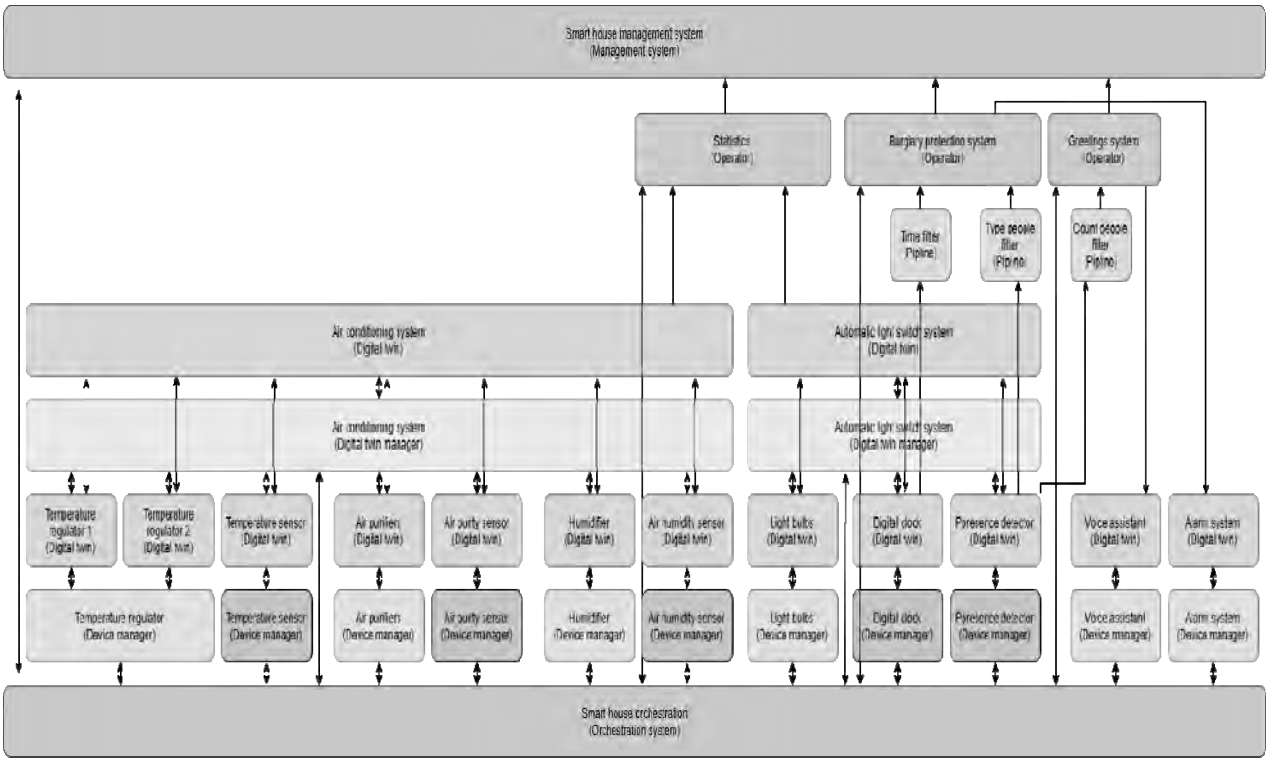

Рис. 8. Діаграма архітектури системи розумного будинку, розробленої відповідно до створеного архітектурного стандарту

Висновки

Ця апаратно-програмна платформа цілком відповідає всім поставленим на початку вимогам сучасної IoT системи:

- легко підтримується;
- за своєю розділеною природою легко масштабується;
- завдяки сучасним контейнерним технологіям є дуже захищеною;
- має високу швидкодію завдяки безсерверним функціям та протоколам зв’язку;
- використовує технології, сертифіковані CNCF та Kubernetes;
- вміє аналізувати дані в режимі реального часу за принципом “hot path”, а концепція Digital Twin дозволяє легко управляти підключеними пристроями.

Також представлена система має простий та зрозумілий стандарт API для побудови програмних IoT-архітектур, дуже легко розгортається на будь-яких одноплатних мінікомп'ютерах із підтримкою Linux.

Підсумовуючи все вищесказане, можна констатувати, що цю систему можна використовувати як платформу для побудови сучасних IoT-рішень за принципом туманних/граничних обчислень.

Список літератури

1. Internet of Things (IoT) Applications for Enterprise Productivity / Koç, Erdinç // IGI Global. 2020. P. 26–28.

2. The Acceleration of Digitization as Result of COVID-19 [Elektronnyy resurs] / Rezhym dostupu: https://www2.deloitte.com/global/en/blog/responsible-business-blog/2020/acceleration-of-digitization-as-result-of-covid-19.html .

3. What is Digital Workspace - Digital Workspace Definition [Elektronnyy resurs] / Rezhym dostupu: https://www.citrix.com/ru-ru/glossary/what-is-digital-workspace.html .

4. The Evolution of the Internet of Things [Elektronnyy resurs] / Rezhym dostupu: https://www.ti.com/lit/ml/swrb028/swrb028.pdf?ts=1606342825763&ref_url=https%253A%252F%252Fwww.google.com%252F .

5. IoT market size worldwide 2017–2025 / Statista [Elektronnyy resurs] / Rezhym dostupu: https://www.statista.com/statistics/976313/global-iot-market-size/ .

6. 2020 International Conference on Applications and Techniques in Cyber / Jemal H. Abawajy, Kim-Kwang Raymond Choo, Zheng Xu, Mohammed Atiquzzaman // Springer Nature 2020. P. 540–541.

7. The IoT Revolution: challenges and opportunities [Elektronnyy resurs] / Rezhym dostupu: https://www.gbnews.ch/the-iot-revolution/ .

8. 5 challenges still facing the Internet of Things – IoT Now – How to run an IoT enabled business [Elektronnyy resurs] / Rezhym dostupu: https://www.iot-now.com/2020/06/03/103228-5-challenges-still-facing-the-internet-of-things/ .

9. The Five Essential IoT Requirements and How to Achieve Them [Elektronnyy resurs] / Rezhym dostupu: https://www.cognizant.com/whitepapers/the-five-essential-iot-requirements-and-how-to-achieve-them-codex4241.pdf.

10. Internet of Things (IoT) Applications for Enterprise Productivity / Koç, Erdinç // IGI Global 2020. P. 9–16.

11. Definition of Edge Computing – Gartner Information Technology Glossary [Elektronnyy resurs] / Rezhym dostupu: https://www.gartner.com/en/information-technology/glossary/edge-computing#:~:text=Edge%20computing%20is%20part%20of,produce%20or%20consume%20that%20information.

12. computing-overview.pdf [Elektronnyy resurs] / Rezhym dostupu: https://www.cisco.com/c/dam/en_us/solutions/trends/iot/docs/computing-overview.pdf.

13. Edge Computing vs. Fog Computing: What's the Difference? [Elektronnyy resurs] / Rezhym dostupu: https://www.cmswire.com/information-management/edge-computing-vs-fog-computing-whats-the-difference/#:~:text=%E2%80%9CFog%20computing%20and%20edge%20computing%20are%20effectively%20the%20same%20thing.&text=So%2C%20with%20Fog%20computing%2C%20the,itself%20without%20being%20transferred%20anywhere.

14. Differences between Cloud, Fog and Edge Computing in IoT – Digiteum [Elektronnyy resurs] / Rezhym dostupu: https://www.digiteum.com/cloud-fog-edge-computing-iot .

15. What is a Container? / App Containerization / Docker [Elektronnyy resurs] / Rezhym dostupu: https://www.docker.com/resources/what-container .

16. Cloud Native Computing Foundation [Електронний ресурс] / Rezhym dostupu: https://www.cncf.io/ .

17. Services for CNCF projects / Cloud Native Computing Foundation [Elektronnyy resurs] / Режим доступу: https://www.cncf.io/services-for-projects/ .

18. CNCF Cloud Native Interactive Landscape [Elektronnyy resurs] / Rezhym dostupu: https://landscape.cncf.io/ .

19. IoT Hardware Guide | 2019 Prototyping Boards & Development Kit Options [Elektronnyy resurs] / Rezhym dostupu: https://www.postscapes.com/internet-of-things-hardware/.

20. Raspberry Pi Microcomputers: the icing on the IoT cake / Indeema Software [Elektronnyy resurs] / Rezhym dostupu: https://indeema.com/blog/raspberry-pi-microcomputers--the-icing-on-the-iot-cake

21. rpi_DATA_2711_1p0_preliminary.pdf [Elektronnyy resurs] / Rezhym dostupu: https://www.raspberrypi.org/documentation/hardware/raspberrypi/bcm2711/rpi_DATA_2711_1p0_preliminary.pdf .

22. K3s: Lightweight Kubernetes [Elektronnyy resurs] / Rezhym dostupu: https://k3s.io/

23. Knative [Elektronnyy resurs] / Режим доступу: https://knative.dev/

24. Nuclio: Serverless Platform for Automated Data Science [Elektronnyy resurs] / Rezhym dostupu: https://nuclio.io/

25. The world's lightest, fastest service mesh. / Linkerd [Elektronnyy resurs] / Rezhym dostupu: https://linkerd.io/

26. NATS – Open Source Messaging System / Secure, Native Cloud Application Development [Elektronnyy resurs] / Rezhym dostupu: https://nats.io/

27. gRPC – A high-performance, open source universal RPC framework [Elektronnyy resurs] / Rezhym dostupu: https://grpc.io/

28. Open Source In-Memory Computing Platform – Apache Ignite® [Elektronnyy resurs] / Rezhym dostupu: https://ignite.apache.org/

29. TAOS Data / Big Data Platform Designed and Optimized for IoT [Електронний ресурс] / Rezhym dostupu: https://www.taosdata.com/en/

30. GitHub-dgraph-io/badger: Fast key-value DB in Go. [Elektronnyy resurs] / Rezhym dostupu: https://github.com/dgraph-io/badger.

31. Implementing Domain-Driven Design / Vaughn Vernon // Academic Press. 2013. P. 1–6.

32. Event-Driven Architecture: How SOA Enables the Real-Time Enterprise / Hugh Taylor, Angela Yochem, Les Phillips, Frank Martinez // Pearson Education. 2009. P. 11–13.

33. Microservices [Elektronnyy resurs] / Rezhym dostupu: https://martinfowler.com/articles/microservices.html .

34. Serverless [Elektronnyy resurs] / Rezhym dostupu: https://martinfowler.com/bliki/Serverless.html .

35. Digital Twin Driven Smart Manufacturing / Fei Tao, Meng Zhang, A.Y.C. Nee // Wesley. 2019. P. 7–10.

DEVELOPMENT OF THE HARDWARE AND SOFTWARE PLATFORM FOR MODERN IOT SOLUTIONS BASED ON FOG COMPUTING USING CLOUD-NATIVE TECHNOLOGIES

A. Shevchenko, S. Puzyrov

Petro Mohyla Black Sea National University, Mykolaiv, Ukraine
Computer Engineering Department

The concept of digital transformation is very relevant at the moment due to the epidemiological situation and the transition of the world to the digital environment. IoT is one of the main drivers of digital transformation. The Internet of Things (IoT) is an extension of the Internet, which consists of sensors, controllers, and other various devices, the so-called "things," that communicate with each other over the network. In this paper, the development of hardware and software for the organization of fog and edge computing was divided into three levels: hardware, orchestration, application. Application level also was divided into two parts: software and architectural. The hardware was implemented using two versions of the Raspberry Pi: Raspberry Pi 4 and Raspberry Pi Zero, which are connected in master-slave mode. The orchestration used K3S, Knative and Nuclio technologies. Technologies such as Linkerd service network, NATS messaging system, implementation of RPC – GRPC protocol, TDengine database, Apache Ignite, Badger were used to implement the software part of the application level. The architecture part is designed as an API development standard, so it can be applied to a variety of IoT software solutions in any programming language. The system can be used as a platform for construction of modern IoT-solutions on the principle of fog/edge computing.

Keywords: Internet of Things, IoT-platform, Container technologies, Digital Twin, API.