УДК 004.42
DOI: 10.31673/2412-9070.2025.042607
О. Б. ПРИДИБАЙЛО, старший викладач;
ORCID: 0009-0003-7967-5827
Р. В. ПРИДИБАЙЛО, аспірант,
ORCID: 0009-0003-1747-7518
Державний університет інформаційно-комунікаційних технологій, Київ

РОЛЬ КОНТЕЙНЕРИЗАЦІЇ У ХМАРНИХ ОБЧИСЛЕННЯХ

Контейнеризація стала важливою технологією для модернізації інфраструктурних рішень в умовах цифрової трансформації, тому доцільно розглянути її роль у хмарних обчисленнях. У даній статті аналізується сприяння контейнеризації для оптимізації процесів розгортання, масштабування та управління застосунками у хмарних середовищах. Для цього розглянуто основні ідеї контейнеризації: принципи роботи контейнерів, їх взаємодія з операційними системами, основні технології, що використовуються для створення та управління контейнерами (наприклад, Docker, Kubernetes, Docker Swarm) та інші інструменти оркестрації. Також у статті описуються переваги контейнеризації: швидке розгортання застосунків, покращена портативність, ефективне використання ресурсів, підтримка мікросервісної архітектури, а також підвищена гнучкість і масштабованість у хмарних середовищах; розглядаються безпека, моніторинг, оркестрація контейнерів та управління складними середовищами з великою кількістю контейнеризованих сервісів.

У статті визначено важливість питань безпеки та складнощі при впровадженні: складність у налаштуванні та підтримці інструментів оркестрації, необхідність адаптації підходів до розробки програмного забезпечення. Стаття пропонує глибоке розуміння розгляду контейнеризації у контексті хмарних обчислень, її впливу на сучасну IT-інфраструктуру та роль в оптимізації бізнес-процесів. Також у статті розглянуто, як контейнеризація змінює підходи до розробки та експлуатації програм, що забезпечує компаніям достатньо високу ефективність, гнучкість і надійність у роботі з хмарними сервісами. Розглянуто роль контейнеризації у хмарних обчисленнях, її основи, переваги та обмеження, а також допомогу оркестрації контейнерів у хмарних середовищах для зменшення складності управління IT-інфраструктурою.

Ключові слова: контейнеризація; хмарні обчислення; docker; kubernetes; оркестрація контейнерів; мікросервісна архітектура; хмарна платформа; портативність застосунків; віртуалізація; сервіс; IT-інфраструктура; моніторинг.

Вступ

Стрімкий розвиток інформаційних технологій, а також зростаючі вимоги до масштабованості, доступності та ефективності обчислювальних систем зумовили активне впровадження хмарних обчислень у різних галузях — від бізнесу до науки та державного управління. Хмарні сервіси дозволяють динамічно розподіляти ресурси, скорочувати витрати на інфраструктуру та оптимізувати процеси розробки й розгортання програмного забезпечення [8, 9].

У процесі еволюції хмарних технологій виникла потреба у гнучкіших і технологічно досконаліших підходах до управління прикладними сервісами. Зокрема, одним із найефективніших рішень на цьому етапі розвитку стало впровадження контейнеризації — методу ізоляції програмного забезпечення та його залежностей у самодостатні контейнери, які можна розгортати у будь-якому середовищі без модифікації початкового коду [2, 4].

Контейнеризація поєднує у собі переваги класичної віртуалізації та легкості в адмініструванні, дозволяючи досягати високого рівня автоматизації, зменшення часу на розгортання сервісів і спрощення підтримки мікросервісної архітектури [3, 6]. Використання таких інструментів, як сервісно-орієнтоване проєктування для створення контейнерів та програмних платформ

для автоматичного управління контейнеризованими застосунками (зокрема, Kubernetes), стало стандартом у галузі розробки та супроводу хмарних застосунків [1, 5].

Актуальність теми зумовлена тим, що контейнеризація поступово стає невід'ємною складовою сучасних хмарних середовищ. Водночас практичне впровадження цієї технології супроводжується низкою викликів, серед яких — забезпечення безпеки, ефективного моніторингу та управління розподіленими контейнеризованими системами [10, 11].

Основна частина

Контейнеризація — це метод упаковування та запуску програмного забезпечення в ізольованому середовищі, яке включає в себе все необхідне для його коректної роботи: код, що виконується, бібліотеки, конфігураційні файли, залежності тощо. Завдяки цьому забезпечується портативність програмного забезпечення між різними середовищами — від локального комп'ютера розробника до хмарної інфраструктури [2, 3].

На відміну від традиційної віртуалізації, де кожна віртуальна машина має власну операційну систему, контейнер працює поверх ядра спільної операційної системи, що дозволяє значно зменшити споживання ресурсів та час запуску [5]. Хоча контейнери та віртуальні машини виконують схожі функції — ізоляцію програмного середовища — підходи, на яких вони базуються, суттєво відрізняються як з архітектурної, так і з практичної точок зору [4, 6].

Основна відмінність полягає у рівні віртуалізації. Віртуальні машини створюють повністю ізольовані екземпляри, кожен із власною операційною системою, що працює поверх гіпервізора. Це забезпечує надійну ізоляцію, але водночас потребує значно більше ресурсів, оскільки кожна віртуальна машина дублює повний стек операційної системи. Як результат — більші затрати на оперативну пам'ять, час запуску та зберігання [7].

Контейнери, навпаки, використовують спільне ядро операційної системи хоста, ізолюючи лише середовище виконання застосунку. Вони не вимагають запуску окремої ОС, що робить їх легшими, швидшими та значно менш ресурсоємними. Завдяки цьому на одному фізичному сервері можна розмістити більше контейнерів, ніж віртуальних машин [1, 3].

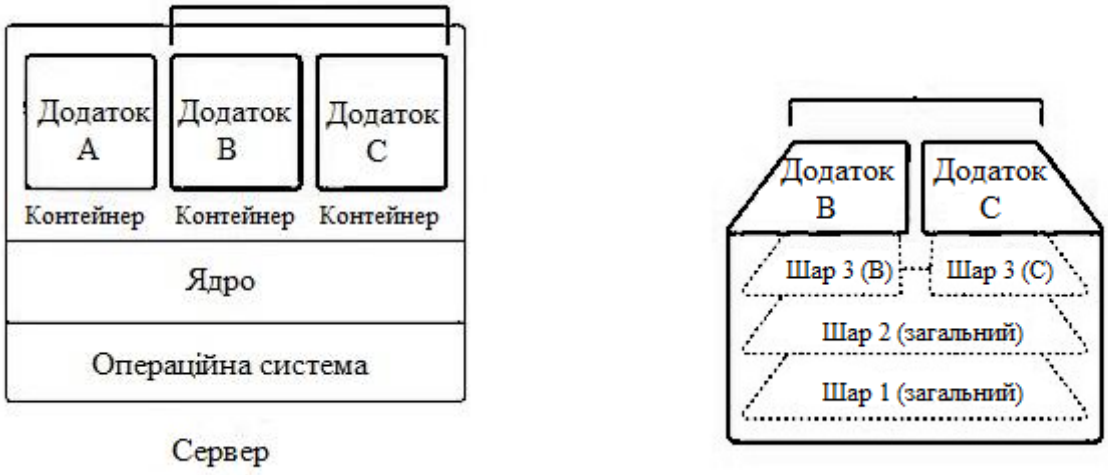

Рис. 1. Загальний вигляд сервера з контейнерами та вміст контейнера

З практичного боку це означає, що контейнери краще підходять для мікросервісної архітектури, CI/CD-процесів (CI (Continuous Integration) — це неперервна інтеграція, а CD (Continuous Delivery) — неперервна доставка) та швидкого масштабування у хмарному середовищі [3, 6]. Віртуальні машини, своєю чергою, залишаються актуальними у випадках, коли потрібна повна ізоляція, робота з різними ОС або запуск "важких" застосунків, що мають високі вимоги до безпеки або сумісності [4, 7].

Ще один аспект — швидкість запуску. Контейнери запускаються майже миттєво, тоді як віртуальні машини потребують більше часу на ініціалізацію. Це критично важливо у динамічних системах, які мають швидко реагувати на зміну навантаження.

Таким чином, контейнери й віртуальні машини — це не взаємовиключні технології, а інструменти, кожен із яких має свої сильні сторони. Вибір між ними залежить від конкретних завдань, вимог до безпеки, гнучкості та ресурсів [1, 3].

Найбільш поширеним інструментом для створення та управління контейнерами є сервісно-орієнтоване проєктування (наприклад, Docker, який надає розробникам зручний спосіб опису середовища застосунку за допомогою Dockerfile, а також можливість централізованого зберігання образів контейнерів через Docker Hub). Контейнерний образ у цьому випадку виступає шаблоном, з якого створюються ізольовані екземпляри програм.

Ще одним важливим аспектом контейнеризації є оркестрація — тобто автоматизоване управління великою кількістю контейнерів. Для цього використовуються програмні платформи (наприклад, Kubernetes), які забезпечують автоматичне масштабування, балансування навантаження, розподіл ресурсів та відновлення контейнерів у разі їхньої відмови. Завдяки цьому організації можуть ефективно керувати складними розподіленими системами, що розгортаються у хмарних середовищах.

Контейнеризація також тісно пов’язана з концепцією мікросервісної архітектури, коли велика система складається з незалежних сервісів, кожен з яких виконує окрему функцію і може бути розгорнутий, оновлений або масштабований окремо. Контейнери ідеально підходять для такої моделі, оскільки забезпечують ізольованість, незалежність та швидку зміну окремих компонентів.

Завдяки цим характеристикам контейнеризація набуває все більшого поширення в індустрії. Вона не лише прискорює цикл розробки і впровадження програмного забезпечення, а й спрощує управління інфраструктурою, забезпечуючи при цьому високу гнучкість, стабільність та масштабованість хмарних рішень.

Контейнеризація поступово зайняла чільне місце у сфері хмарних обчислень, оскільки дозволяє вирішувати як суто технічні завдання, так і організаційні виклики, пов’язані з розробкою та підтримкою програмних рішень. У хмарному середовищі контейнери стали зручним інструментом, що дає змогу краще керувати ресурсами, скорочувати час між етапами життєвого циклу застосунку та спрощувати процес масштабування системи під реальні навантаження [1, 6, 8].

Контейнеризація відіграє ключову роль у підвищенні ефективності хмарних обчислень, надаючи низку переваг, які роблять її оптимальним рішенням для побудови масштабованих, гнучких та стійких IT-систем [1, 3, 6], тому перевагами контейнеризації у хмарному середовищі є:

1. Портативність

Однією з ключових переваг використання контейнерів є їхня здатність зберігати працездатність застосунку незалежно від середовища, у якому він запускається. Завдяки тому, що контейнер містить усі необхідні для роботи компоненти — бібліотеки, налаштування, залежності — розробник може бути впевнений, що застосунок працюватиме однаково стабільно як на його власному комп’ютері, так і на хмарному сервері, незалежно від платформи. Це практично усуває типову проблему несумісності між середовищем розробки та розгортання. Це значно спрощує перенесення застосунків між різними середовищами та унеможливлює проблему "працює у мене, але не працює на сервері" [2, 5].

2. Легка масштабованість

Контейнери запускаються і зупиняються значно швидше, ніж традиційні віртуальні машини, що особливо зручно для хмарного середовища з мінливим навантаженням. Це дає змогу оперативно масштабувати застосунок у разі потреби — як за рахунок збільшення доступних ресурсів, так і шляхом розгортання додаткових копій сервісу. Завдяки таким інструментам, як Kubernetes, процес масштабування відбувається автоматично: система реагує на зростання трафіку, піднімаючи нові контейнери, що дозволяє підтримувати стабільну роботу навіть під час різких стрибків навантаження [6, 8].

3. Ефективне використання ресурсів

На відміну від віртуальних машин, кожна з яких запускає власну операційну систему, контейнери спільно використовують ядро тієї ОС, на якій працюють. Завдяки цьому вони займають менше ресурсів і легші у запуску. Це дає змогу розміщувати значно більше контейнерів на одному сервері, що не лише економить обчислювальні потужності, а й підвищує ефективність використання інфраструктури [3, 4].

4. Швидке розгортання і CI/CD

Цей набір методик дозволяє розробникам частіше і надійніше розгортати зміни у програмному забезпеченні.

Контейнеризація суттєво полегшує автоматизацію розгортання застосунків. Вона добре поєднується з підходами DevOps і CI/CD, дозволяючи швидко доставляти оновлення без переривання роботи сервісу. Завдяки цьому можна впроваджувати нові версії через стратегії, коли зміни спочатку отримує лише частина користувачів. Це знижує ризики й дає можливість виявити проблеми ще до повного впровадження [2, 6].

5. Ізоляція та безпека

Кожен контейнер виконується в ізольованому середовищі, що підвищує безпеку системи. Помилки або збої в одному контейнері не впливають на інші. Крім того, сучасні інструменти дозволяють додатково контролювати доступ до мережі, обмежувати привілеї контейнерів та проводити сканування на вразливості ще до їх запуску [1, 7].

6. Гнучкість у керуванні мікросервісами

Контейнеризація ідеально підходить для реалізації мікросервісної архітектури, у якій застосунок складається з окремих компонентів (сервісів), що можуть розгортатися, оновлюватися і масштабуватися незалежно один від одного. Це спрощує супровід великих проєктів, підвищує стійкість до збоїв і прискорює розробку нових функцій [3, 8].

Попри численні переваги, контейнеризація у хмарному середовищі має і свої виклики, які потребують ретельного врахування під час розробки та впровадження ІТ-рішень:

1. Безпека контейнерів

Оскільки контейнери використовують спільне ядро операційної системи з хост-машиною, це потенційно відкриває шлях до вразливостей. У разі недостатньої ізоляції процесів або неналежного контролю доступу може виникнути ризик витоку даних, ескалації привілеїв або виконання шкідливого коду в межах системи. Щоб мінімізувати ці загрози, необхідно впроваджувати чіткі політики безпеки, використовувати лише перевірені образи з надійних реєстрів (*trusted registries*), а також регулярно оновлювати контейнерні образи з урахуванням останніх патчів і рекомендацій з безпеки [1, 6].

2. Моніторинг і логування

У контейнеризованих середовищах, особливо в системах із мікросервісною архітектурою, завдання моніторингу та збирання логів ускладнюється динамікою та коротким життєвим циклом контейнерів. Традиційні засоби моніторингу, орієнтовані на статичні сервери, часто не здатні повноцінно відслідковувати стан подібних систем. Тому виникає потреба у впровадженні спеціалізованих рішень, які можуть адаптуватися до швидкоплинної природи контейнерів [2, 4, 6].

Наприклад, Prometheus (для метрик), Grafana (для візуалізації), ELK Stack (ElasticSearch, Logstash, Kibana — для централізованого логування) або OpenTelemetry (для трасування та метрик) дозволяють виявляти помилки і вузькі місця та сприяють підвищенню загальної стабільності та керованості контейнерної інфраструктури.

3. Оркестрація та управління

Коли кількість контейнерів зростає, з'являється потреба в інструментах оркестрації. Найпопулярнішим рішенням є програмні платформи, однак їх налаштування та підтримка вимагають глибоких технічних знань. Це ускладнює перші етапи впровадження для невеликих команд або організацій з обмеженими ресурсами. Оркестрація контейнерів — це процес автоматизованого управління контейнерами у хмарних середовищах. Вона дозволяє ефективно контролювати велике число контейнеризованих застосунків, автоматизуючи процеси їх запуску, масштабування та підтримки безперервної роботи. Однією з важливих можливостей оркестра-

ції є самовідновлення. Якщо один з контейнерів виходить з ладу, оркестратор автоматично запустить новий, щоб зберегти доступність сервісів. Також оркестрація дозволяє застосовувати різні стратегії розгортання нових версій застосунків [5, 7].

Оркестрація контейнерів у хмарі

Одним з основних інструментів для оркестрації контейнерів є Kubernetes. Це система, яка допомагає управляти контейнерами на кількох серверах або хмарних платформах, автоматично масштабуючи їх в залежності від потреб. Дана програмна платформа також забезпечує балансування навантаження, тобто розподіляє трафік між контейнерами, щоб уникнути перевантаження [1, 12].

Ще одна важлива складова оркестрації — це моніторинг та логування. Завдяки цим інструментам, які інтегруються з оркестраторами, можна відслідковувати продуктивність контейнерів, виявляти помилки і проблеми, що виникають у процесі роботи.

Завдяки оркестрації, організації можуть створювати високонадійні та масштабовані системи, що автоматично підлаштовуються під змінювані умови навантаження. Ця технологія ідеально підходить для великих, складних і динамічних хмарних середовищ [11].

Майбутнє контейнеризації у хмарних обчисленнях

Контейнеризація вже давно стала невід'ємною частиною хмарних технологій і в майбутньому її роль лише зростатиме. Завдяки своїй гнучкості та ефективності, контейнери допомагають вирішувати завдання швидкого масштабування та автоматизації, і ці можливості будуть тільки розширюватися.

Одним з основних напрямків розвитку контейнеризації стане поглиблене використання оркестраторів. Ці системи вже зараз дозволяють автоматично керувати контейнерами, знижуючи навантаження на адміністраторів і підвищуючи стабільність роботи застосунків. У майбутньому програмні платформи та подібні технології ще більше автоматизуватимуть управління контейнерами, що дозволить знижувати витрати на інфраструктуру і покращувати масштабованість застосунків.

Іншим важливим аспектом є покращення безпеки контейнерів. З кожним роком зростає кількість загроз і вразливостей, тому розробники будуть впроваджувати нові методи захисту, зокрема більш складні системи шифрування, контроль доступу та автоматичне сканування контейнерних образів на наявність шкідливого коду. Це допоможе зробити контейнери більш безпечними для зберігання даних та запуску застосунків у хмарі.

Ще одним перспективним напрямком є розвиток *serverless* технологій (*serverless computing* (безсерверні обчислення) - це підхід до розробки програмного забезпечення, при якому розробники можуть створювати програми, зберігати дані та налаштовувати інтеграцію з іншими платформами без необхідності створення віртуальних машин або обслуговування інфраструктури) у поєднанні з контейнерами. Завдяки цьому контейнери можуть стати ідеальним середовищем для безсерверних обчислень, де ресурси виділяються лише за потребою, що дозволяє значно оптимізувати витрати на інфраструктуру [1, 5].

Також контейнеризація стане ще важливішою у створенні складних архітектур, де кілька контейнерів взаємодіють між собою. Це буде корисно для таких технологій, як обробка великих даних, Інтернет речей (IoT) або штучний інтелект. Завдяки такій гнучкості можна буде будувати високопродуктивні системи, які швидко адаптуються до змінюваних умов.

Загалом, контейнеризація має велике майбутнє у хмарних обчисленнях. Вона буде ставати ще більш потужним інструментом для компаній, що прагнуть забезпечити ефективність, гнучкість та безпеку своїх IT-інфраструктур у хмарному середовищі.

Висновки

Контейнеризація є однією з найважливіших технологій, що визначає майбутнє хмарних обчислень. Завдяки своїм численним перевагам, таким як портативність, швидкість запуску, масштабованість та ефективне використання ресурсів — контейнери дозволяють оптимізувати роботу застосунків і суттєво знижують витрати на інфраструктуру. Вони стають основою для створення гнучких і високопродуктивних хмарних систем, що легко адаптуються до змі-

нюваних умов і вимог бізнесу. Контейнеризація сприяє інтеграції з інструментами автоматизації, такими як DevOps і CI/CD, що забезпечує швидкий і безпечний процес розгортання нових версій застосунків. Оркестрація контейнерів, зокрема за допомогою таких платформ, як Kubernetes, дозволяє автоматично масштабувати сервіси і зберігати стабільність системи за будь-яких умов. Крім того, оркестратори надають можливість автоматизувати відновлення контейнерів у разі збою, що забезпечує безперервну роботу застосунків, навіть у разі аварій. Попри те, що контейнеризація приносить чимало переваг, вона також вимагає уваги до питань безпеки, моніторингу та управління. Правильне налаштування та використання інструментів для контролю доступу, сканування контейнерних образів та інші заходи безпеки є необхідними для зменшення ризиків. Без ефективного моніторингу і логування проблеми із запуском контейнерів, некоректними версіями або збої в інфраструктурі можуть бути важко виявлені. Впровадження спеціалізованих рішень, таких як Prometheus або ELK Stack, є критично важливим для забезпечення стабільності і прозорості роботи контейнеризованих застосунків.

У майбутньому, з розвитком технологій, контейнеризація стане ще більш інтегрованою - з іншими інноваційними підходами, такими як мультихмарні стратегії, безсерверні обчислення та інтеграція зі штучним інтелектом для автоматизованого управління інфраструктурою. Це дозволить досягати ще більшої ефективності і зменшення витрат на обчислювальні ресурси при одночасному забезпеченні високої доступності та продуктивності. Загалом, контейнеризація сприяє значному підвищенню ефективності хмарних обчислень, роблячи їх більш гнучкими, масштабованими та економічно вигідними. Це технологія, яка вже зараз змінює підходи до розробки, впровадження та управління застосунками і в майбутньому її роль лише зростатиме. Враховуючи тенденції до більшої автоматизації та розвитку хмарних інфраструктур, контейнеризація має стати основою для побудови сучасних, високопродуктивних і надійних обчислювальних систем.

Список літератури

1. Burns B., Grant B., Oppenheimer D., Brewer E., Wilkes J. Borg, Omega, and Kubernetes: Lessons learned from three container-management systems over a decade // ACM Queue. – 2016. – Vol. 14, No. 1. – C. 70–93.

2. Merkel D. Docker: Lightweight Linux containers for consistent development and deployment // Linux Journal. – 2014. – No. 239. – C. 2–6.

3. Pahl C. Containerization and the PaaS cloud // IEEE Cloud Computing. – 2015. – Vol. 2, No. 3. – C. 24–31.

4. Turnbull J. The Docker Book: Containerization is the new virtualization. – 2nd ed. – San Francisco: James Turnbull, 2014. – 328 с.

5. Hightower K., Burns B., Beda J. Kubernetes: Up and Running. – 2nd ed. – Sebastopol, CA: O'Reilly Media, 2017. – 238 с.

6. Mouat A. Using Docker: Developing and Deploying Software with Containers. – Sebastopol, CA: O'Reilly Media, 2015. – 351 с.

7. Stallman R., Lessig L. Free Software, Free Society: Selected Essays of Richard M. Stallman. – 3rd ed. – GNU Press, 2020. – 430 с.

8. Романюк О.Ю. Технології хмарних обчислень у сучасних інформаційних системах // Наукові записки. – 2021. – № 5. – С. 145–152.

9. Ярошенко С.В. Контейнеризація застосунків в хмарних середовищах: огляд сучасних підходів // Інформаційні технології та комп'ютерна інженерія. – 2020. – № 1 (15). – С. 38–46.

10. IT-бізнес в Україні: огляд ринку хмарних технологій / [Електронний ресурс]. – Режим доступу: https://itukraine.org.ua/cloud_market_report.html

11. Red Hat. Understanding containers and container orchestration [Електронний ресурс]. – Режим доступу: https://www.redhat.com/en/topics/containers/what-is-container-orchestration

12. Google Cloud Documentation. Containers and Kubernetes [Електронний ресурс]. – Режим доступу: https://cloud.google.com/containers

13. Cloud Native Computing Foundation. Cloud Native Landscape [Електронний ресурс]. – Режим доступу: https://landscape.cncf.io

O. Prydybailo, R. Prydybailo

THE ROLE OF CONTAINERIZATION IN CLOUD COMPUTING

Containerization has become an important technology for modernizing infrastructure solutions in the context of digital transformation, so it is appropriate to consider its role in cloud computing. This article analyzes the contribution of containerization to optimizing the processes of deployment, scaling, and management of applications in cloud environments. To do this, the main ideas of containerization are considered: the principles of container operation, their interaction with operating systems, the main technologies used to create and manage containers (Docker, Kubernetes, Docker Swarm) and other orchestration tools. The article also describes the advantages of containerization: rapid deployment of applications, improved portability, efficient use of resources, support for microservice architecture, as well as increased flexibility and scalability in cloud environments; security, monitoring, orchestration of containers, and management of complex environments with a large number of containerized services are considered. The article highlights the importance of security issues and implementation challenges: the complexity of configuring and maintaining orchestration tools, the need to adapt software development approaches. The article offers a deep understanding of containerization in the context of cloud computing, its impact on modern IT infrastructure and its role in optimizing business processes. The article also discusses how containerization changes approaches to developing and operating applications, providing companies with sufficiently high efficiency, flexibility and reliability in working with cloud services.

This article discusses the role of containerization in cloud computing, its foundations, advantages and limitations, as well as the help of container orchestration in cloud environments to reduce the complexity of managing IT infrastructure.

Keywords: containerization; cloud computing; docker; kubernetes; container orchestration; microservice architecture; cloud platform; application portability; virtualization; service; IT infrastructure; monitoring.