TECHNICAL SCIENCES

CROSS-PLATFORM SOLUTIONS IN INFORMATION SYSTEMS: OPPORTUNITIES AND CHALLENGES OF INTEGRATION

КРОСПЛАТФОРМНІ РІШЕННЯ В ІНФОРМАЦІЙНИХ СИСТЕМАХ: МОЖЛИВОСТІ ТА ВИКЛИКИ ІНТЕГРАЦІЇ

Olha Androshchuk[1]
Hanna Lytovchenko[2]

DOI: https://doi.org/10.30525/978-9934-26-536-5-25

У сучасному світі кросплатформні рішення дозволяють організаціям забезпечити безперервність процесів, гнучкість у виборі технологічних платформ та ефективну інтеграцію різних цифрових сервісів. Завдяки підтримці різних операційних систем і архітектур, такі рішення сприяють розгортанню програмного забезпечення, мінімізують витрати на його адаптацію та прискорюють впровадження нових технологій. Особливо важливим є їх застосування в хмарних середовищах, де забезпечується взаємодія між різними інфраструктурними платформами.

Попит на кросплатформні рішення за умови швидкої масштабованості, підтримки мобільності та забезпечення єдиного користувальницького досвіду незалежно від пристрою чи операційної системи. Використання стандартизованих API, контейнеризації та оркестрації дозволяє компаніям і державним установам створити стійкі, ефективні та безпечні інформаційні системи.

Широке впровадження кросплатформних технологій сприяє підвищенню доступності цифрових сервісів та ефективності їх використання. Це особливо важливо в умовах динамічного розвитку інформаційних технологій, коли необхідно забезпечити швидку адаптацію систем до нових вимог і стандартів. Використання контейнеризації, API та гібридних архітектур значно спрощує інтеграцію різнорідних компонентів і покращує керованість інформаційної інфраструктури.

Кросплатформність, полегшує інтеграцію різних технологічних рішень, забезпечує сумісність між хмарними провайдерами та підтримує

[1] National Defence University of Ukraine Centre for Military and Strategic Studies, Ukraine
[2] National Defence University of Ukraine Centre for Military and Strategic Studies, Ukraine

гнучкість у виборі інфраструктури. Це дає змогу знизити ризики залежності від конкретного постачальника (vendor lock-in) та адаптувати інформаційні системи до різних сценаріїв використання.

Кросплатформність (Cross-Platform) – це здатність програмного забезпечення, апаратних систем або мережевих рішень функціонувати на різних операційних системах, пристроях або технологічних середовищах без втрати продуктивності та функціональності.

Кросплатформність – властивість програмного забезпечення працювати більш ніж на одній програмній або апаратній платформі; технології, що дозволяють досягти такої властивості. Кросплатформність дозволяє суттєво скоротити витрати на розробку нового або адаптацію існуючого програмного забезпечення [1].

Кросплатформність у мережевих та хмарних інфраструктурах забезпечує інтеграцію різних сервісів незалежно від їхнього середовища виконання, що є критичним для мультихмарних рішень (AWS, Azure, GCP). У військовій сфері вона реалізується через системи SitaWare Headquarters, JADC2, IVAS, сприяючи координації операцій та взаємодії розрізнених систем [2].

До ключових технологій кросплатформного підходу належать API, контейнеризація та віртуалізація. API стандартизує взаємодію між застосунками та сервісами, забезпечуючи узгоджену комунікацію без залежності від реалізації. Інтерфейси можуть бути представлені веб-сервісами (REST, SOAP), бібліотеками або системними викликами, що дозволяє інтегрувати різні платформи та середовища [3].

Контейнерізація забезпечує ізоляцію застосунків у компактних середовищах (контейнерах), які використовують спільне ядро операційної системи, що підвищує ефективність використання ресурсів та забезпечує портативність. Це мінімізує витрати, прискорює розгортання та спрощує управління розподіленими системами, зокрема завдяки інструментам оркестрації, таким як Kubernetes.

Оркестрація – це програмна технологія для автоматичного керування контейнерами. А самі контейнери є середовищами для виконання ізольованих процесів, файли для запуску яких знаходяться в одному образі. Контейнери не пов’язані з апаратною частиною обладнання, тому їх легко переносити між фізичними машинами, хмарами та операційними системами [4].

Kubernetes та контейнери нерозривно пов’язані з розробкою. Саме контейнеризація дозволяє розробникам ізолювати застосунки в різних середовищах без або з мінімальними змінами, роблячи процес розробки, розгортання та керування ними простішим, гнучкішим та зрозумілішим.

Kubernetes підтримує такі контейнери: Docker, containerd, CRI-O та інші з інтерфейсом CRI (Container Runtime Interface) [5].

Віртуалізація дозволяє створювати віртуальні машини (ВМ), кожна з яких має власну операційну систему та працює незалежно від інших. Це забезпечує високий рівень ізоляції, підвищує безпеку та ефективно розподіляє обчислювальні ресурси через гіпервізор. Вона широко використовується в хмарних інфраструктурах і дата-центрах, оптимізуючи використання апаратних ресурсів та спрощуючи управління середовищем.

Основна різниця між віртуалізацією та контейнерізацією полягає в рівні ізоляції: контейнери працюють у межах спільного ядра ОС, тоді як віртуальні машини мають власні операційні системи. Завдяки поєднанню API, контейнерізації та віртуалізації можна створювати гнучкі та масштабовані рішення, які ефективно працюють у мультихмарних і гібридних інфраструктурах.

Кросплатформність дозволяє створювати додатки та сервіси, які можуть працювати на різних платформах, незалежно від операційної системи або апаратного забезпечення. Це знижує витрати на розробку, оскільки один і той самий код може бути використаний для різних середовищ (Windows, Linux, macOS, мобільні платформи). Крім того, це забезпечує гнучкість при виборі технологічних рішень, що дозволяє організаціям швидко адаптуватися до нових умов і вимог. Кросплатформність також підвищує доступність програм, забезпечуючи підтримку ширшого кола користувачів і пристроїв.

Незважаючи на численні переваги, кросплатформність має і свої недоліки. Перш за все, універсальні рішення можуть бути менш ефективними порівняно з програмами, розробленими спеціально для однієї платформи. Це пов’язано з необхідністю підтримки сумісності між різними операційними системами та пристроями. Важливою проблемою є також обмеження в функціональності: деякі специфічні можливості платформи можуть бути недоступними через використання кросплатформних інструментів. Також, у разі значних змін в одній із платформ, розробникам може знадобитися оновлення чи адаптація коду для забезпечення сумісності. Більш детально переваги і недоліки кросплатформного підходу можна подивитись у таблиці 1.

Кросплатформенність є ідеальним рішенням для організацій, які потребують швидкого виходу на різні платформи з мінімальними витратами, але вона не завжди підходить для розв’язання складних технічних завдань, що вимагають високої продуктивності або особливу функціональність. Слід ретельно обирати підхід для кожного конкретного випадку, враховуючи обмеження та специфіку використовуваних

платформ. Використання таких інструментів, як Docker, Kubernetes, або використання кросплатформних фреймворків може значно полегшити задачу забезпечення сумісності та підтримки різних середовищ.

Таблиця 1

Переваги і недоліки впровадження кросплатформного підходу

Вимоги	Переваги	Недоліки
Гнучкість	Зменшення витрат на розробку за рахунок запуску на різних операційних системах	Постійні оновлення і адаптації коду для підтримки на нових версіях платформ
Продуктивність	Оптимізація витрат на розробку та підтримку коду за рахунок універсального коду для всіх платформ	Універсальний код може бути менш ефективним відносно з оптимізованим для конкретної платформи
Покриття	Доступність додатків для усіх користувачів на усіх видах пристроїв	Обмеження в доступі до спеціальних можливостей і функціоналу окремих платформ
Адаптивність	Легкість адаптації до нових платформ	Специфічні вимоги дуже складно реалізувати під конкретні платформи

Джерело: розроблено авторами

Отже, кросплатформність забезпечує створення уніфікованих рішень, які можуть функціонувати в різних хмарних середовищах та апаратних платформах. Завдяки використанню API, контейнеризації (Docker, Kubernetes) та технологій віртуалізації забезпечується взаємодія між сервісами та провайдерами. Це знижує залежність від конкретних постачальників хмарних послуг (vendor lock-in) і підвищує гнучкість у розподілі обчислювальних ресурсів.

References:

1. Багатоплатформність / Вікіпедія. URL: https://uk.wikipedia.org/wiki/Багатоплатформність (дата звернення: 17.03.2025).
2. Why (and when) you should use Kubernetes. URL: https://hackernoon.com/why-and-when-you-should-use-kubernetes8b50915d97d8 (дата звернення: 17.03.2025).
3. Kubernetes vs. Docker: A Primer. URL: https://blog.colobridge.net/uk/2023/12/kubernetes-what-is-it-ua/ (дата звернення: 17.03.2025).
4. Kubernetes vs. Docker: A Primer. URL: https://aws.amazon.com/ru/compare/the-difference-between-kubernetes-and-docker/ (дата звернення: 17.03.2025).
5. Kubernetes: що це таке? URL: https://blog.colobridge.net/uk/2023/12/kubernetes-what-is-it-ua/ (дата звернення: 17.03.2025).