

## Лекція 2

Тема: «Сучасні підходи до управління проектами: традиційні і гнучкі»



# План

1. Гнучкі методології управління проєктами та їх відмінності від традиційних методологій
2. Методологічні засади Scrum-системи
3. Методологічні засади Kanban-системи
4. Цифрові системи управління проєктами в умовах застосування гнучких підходів

# 1. Гнучкі методології управління проєктами та їх відмінності від традиційних методологій

Традиційний підхід ґрунтується на моделі **Waterfall (каскадна модель)**, запропонованій у 1970 р. У. Ройсом, і на процесній логіці стандарту **PMBOK (Project Management Body of Knowledge)** Інституту управління проєктами (PMI).

**Каскадна модель (waterfall model)**



## Основні характеристики каскадної моделі:

- ➔ **Лінійно-послідовна структура.** Проєкт проходить чітко визначені фази: ініціація, планування, виконання, моніторинг і контроль, завершення. Перехід до наступної фази відбувається лише після повного завершення попередньої.
- ➔ **Детальне попереднє планування** (обсяг робіт (score), терміни, бюджет і ресурси) фіксуються на початку проєкту й розглядаються як відносно стабільні (зміст – час – вартість).
- ➔ **Формалізована документація:** технічне завдання, статут проєкту, ієрархічна структура робіт (WBS), календарний план (діаграма Ганта).Централізоване управління: керівник проєкту одноосібно приймає ключові рішення, розподіляє завдання та контролює виконання.
- ➔ **Орієнтація на відповідність плану.** Успіх проєкту оцінюється за критерієм «виконано в межах бюджету, термінів і початкового обсягу».

**!** *Такий підхід ефективний у проєктах з передбачуваними вимогами, стабільним середовищем і низьким рівнем невизначеності (будівництво, виробництво, інфраструктурні проєкти).*

## Передумови виникнення гнучких підходів в управлінні проєктами

У 1990-2000-х рр. у сфері розробки програмного забезпечення накопичилася критична маса проблем каскадного підходу:

- висока частка проєктів, що завершувалися з перевищенням бюджету/термінів або не відповідали вимогам замовника;
- зростання швидкості змін ринкового середовища та вимог клієнтів;
- неможливість повністю уніфікувати вимоги на старті складних, інноваційних проєктів.

У лютому 2001 р. 17 фахівців з розробки ПЗ сформулювали «Маніфест гнучкої розробки програмного забезпечення» (Agile Manifesto URL: <https://agilemanifesto.org/iso/uk/manifesto.html>), що визначив:

#### 4 ключові цінності:

1. Люди та взаємодія – важливіші за процеси та інструменти.
2. Працюючий продукт – важливіший за вичерпну документацію.
3. Співпраця із замовником – важливіша за узгодження умов контракту.
4. Готовність до змін – важливіша за дотримання початкового плану.

Ці цінності конкретизовано у **12 принципах Agile**, серед яких: пріоритет задоволення клієнта через ранню й безперервну поставку цінного продукту; готовність змінювати вимоги навіть на пізніх стадіях; часта поставка працюючого продукту (від кількох тижнів до кількох місяців); щоденна співпраця бізнесу та розробників; самоорганізовані команди; регулярна рефлексія команди щодо підвищення ефективності.

## Порівняльна характеристика традиційного та гнучкого підходів

| Критерій                      | Традиційний (Waterfall)                                          | Гнучкий (Agile)                                        |
|-------------------------------|------------------------------------------------------------------|--------------------------------------------------------|
| Структура процесу             | Лінійно-послідовна, фазова                                       | Ітеративно-інкрементальна                              |
| Ставлення до змін вимог       | Зміни небажані, контролюються через процедуру управління змінами | Зміни очікувані та вітаються навіть на пізніх етапах   |
| Планування                    | Детальне, на весь проєкт наперед                                 | Адаптивне, короткостроковими циклами                   |
| Роль замовника                | Залучений переважно на старті та в контрольних точках            | Постійна, щоденна взаємодія                            |
| Документація                  | Вичерпна, формалізована                                          | Мінімально достатня                                    |
| Організація команди           | Ієрархічна, функціональна                                        | Самоорганізована, крос-функціональна                   |
| Управління                    | Централізоване (менеджер проєкту)                                | Розподілене (команда + власник продукту + фасилітатор) |
| Критерій успіху               | Відповідність первинному плану                                   | Цінність для замовника, здатність адаптуватися         |
| Поставка результату           | Один раз, наприкінці проєкту                                     | Регулярно                                              |
| Оптимальна сфера застосування | Стабільні, добре визначені проєкти                               | Проекти з високою невизначеністю та мінливістю вимог   |

# Гібридний підхід

На практиці все частіше застосовуються **гібридні моделі** (Agile-Waterfall Hybrid, PRINCE2 Agile, Disciplined Agile), які поєднують формалізоване управління на рівні портфеля бізнесу з гнучким виконанням на рівні команди. Це зумовлено тим, що не всі складові великого проєкту однаково потребують гнучкості (наприклад, регуляторна документація може вимагати каскадного підходу, тоді як розробка продукту вимагає використання ітеративного).

# Переваги гібридного підходу

**1. Баланс між контролем та гнучкістю.** Бізнес на рівні керівництва (портфеля проєктів) потребує передбачуваності, а саме чітких бюджетів, строків, фінансової звітності та стратегічного планування, що є основою Waterfall. Проте командам розробників для створення інноваційного продукту потрібен простір для експериментів і швидка адаптація до змін, що забезпечує Agile.

**2. Регуляторні обмеження.** Неможливо розробляти медичне обладнання, банківські системи чи авіаційне ПЗ виключно за принципом «робимо ітераціями і дивимося, що вийде». Юридичні вимоги, ліцензування, безпека та державні стандарти (compliance) вимагають жорсткого документування, що робиться за каскадною моделлю.

**3. Прагматизм замість фанатизму.** Жодна методологія у чистому вигляді не є ідеальною для масштабних проєктів. Поєднання підходів (наприклад, у рамках фреймворків PRINCE2 Agile або Disciplined Agile) дозволяє використовувати сильні сторони обох підходів, нівелюючи їхні недоліки.

## Недолік гібридного підходу

Гібридний підхід значно складніший в реалізації. Якщо процеси налаштовані неправильно, виникає хаос (так званий «Water-Scrum-Fall»), де менеджмент вимагає жорстких дедлайнів, а розробники «прикривають» затримки «гнучкістю»

## 2. Методологічні засади Scrum-системи

**Scrum** — це найпоширеніший фреймворк (інфраструктура програмних рішень, що полегшує розробку складних систем) гнучкого управління, розроблений Дж. Сазерлендом і К. Швабером (перше офіційне визначення у 1995 р.).

Слово **Scrum** запозичене з регбі (термін *scrummage* означає сутичку навколо м'яча), де для перемоги потрібна злагоджена, щільна командна робота, де кожен гравець чітко знає свою роль і підстраховує іншого.

Головна ідея Scrum — **поділити велику роботу на маленькі шматочки і виконувати їх короткими циклами.**

Scrum є **емпіричним каркасом (framework)**, що спирається на три опори емпіричного управління процесом (заснований на практичному досвіді):

- 1. Прозорість (Transparency)** — усі аспекти процесу видимі для тих, хто відповідає за результат.
- 2. Інспекція (Inspection)** — регулярна перевірка артефактів і прогресу для виявлення відхилень.
- 3. Адаптація (Adaptation)** — коригування процесу за результатами інспекції.

**Цінності Scrum, такі як відкритість, повага, сміливість, фокус, відданість, формують культурне підґрунтя команди проєкту.**

## **Ролі в Scrum**

- **Власник продукту (Product Owner)** — відповідає за максимізацію цінності продукту, управляє беклогом продукту, визначає пріоритети.
  - **Scrum-майстер (Scrum Master)** — фасилітатор процесу, (фасилітація – це процес організації групової взаємодії, спрямований на отримання продуктивного результату) відповідає за розуміння й застосування Scrum командою та організацією, усуває перешкоди.
  - **Команда розробки (Developers)** — крос-функціональна самоорганізована група, яка створює інкремент продукту в кожному спринті (Інкремент продукту – це **готова, протестована та робоча версія продукту**, яку команда створила за один спринт).
- Разом ці три ролі утворюють **Scrum-команду**, чисельність якої зазвичай складає 10 і менше осіб.

## Артефакти Scrum

**Артефакти Scrum** — матеріальні та видимі результати або інформація, що створені для забезпечення прозорості, перевірки та адаптації робочого процесу. Згідно з офіційним Посібником зі Скрам, існує **три основні артефакти**:

- 1. Беклог продукту (Product Backlog)** — впорядкований, живий перелік усього, що потрібно для продукту; його зобов'язання — **ціль продукту (Product Goal)**.
- 2. Беклог спринту (Sprint Backlog)** — набір елементів беклогу продукту, обраних для спринту, разом із планом їх реалізації; зобов'язання — **ціль спринту (Sprint Goal)**.
- 3. Інкремент (Increment)** — конкретний робочий результат, сума всіх елементів беклогу продукту, які були виконані протягом поточного та попередніх спринтів. Кожен інкремент має відповідати Визначенню готовності (Definition of Done) та бути придатним до використання.

## Події Scrum

Усі події обмежені в часі і відбуваються в межах **спринту** — базового циклу тривалістю **не більше одного місяця** (типово від 1 до 4 тижнів), протягом якого створюється придатний до використання інкремент (результат).

- 1. Планування спринту (Sprint Planning)** — відповідає на питання: чому цінний цей спринт, що можна зробити, як буде виконано роботу.
- 2. Щоденний скрам (Daily Scrum)** — 15 хв щодня, синхронізація команди розробки, інспекція прогресу до цілі спринту.
- 3. Огляд спринту (Sprint Review)** — до 4 год; демонстрація інкременту зацікавленим сторонам, отримання зворотного зв'язку, адаптація беклогу продукту.
- 4. Ретроспектива спринту (Sprint Retrospective)** — до 3 год; аналіз того, як пройшов спринт з погляду людей, взаємодій, процесів та інструментів, планування покращень.

### 3. Методологічні засади Kanban-системи

**Kanban** (з япон. – «візуальна картка», «сигнал») – метод управління потоком роботи, що бере початок з виробничої системи Toyota (Т. Оно, 1940–1950-ті рр.) і був адаптований для інтелектуальної та проєктної діяльності Д. Андерсоном (книга David J. Anderson. *Kanban: Successful Evolutionary Change for Your Technology Business*, 2010. Це відома праця *Девіда Андерсона (David J. Anderson)*, яку часто називають «біблією Канбану». Автор адаптував принципи виробничої системи Toyota (Lean) для сфери розробки програмного забезпечення та інтелектуальної праці).

На відміну від Scrum, Kanban є **методом еволюційних змін**, а не фреймворком з фіксованими ролями та подіями: він застосовується поверх наявного процесу без обов'язкової його перебудови.

## Базові принципи Kanban

1. Починати з того, що є зараз (не змінюйте процес радикально від початку).
2. Погоджуватися на еволюційні, поступові зміни.
3. Поважати наявні ролі, відповідальність і посади.
4. Заохочувати лідерство на всіх рівнях.

# Практична реалізація Kanban

1. **Візуалізація потоку робіт** — за допомогою канбан-дошки з колонками, що відповідають етапам процесу.
2. **Візуалізація потоку робіт** — за допомогою канбан-дошки з колонками, що відповідають етапам процесу (наприклад: «Беклог» → «В роботі» → «Тестування» → «Готово»).
3. **Обмеження незавершеної роботи** (WIP-ліміти, Work-in-Progress limits) — встановлення максимальної кількості завдань, які можуть одночасно перебувати в певному етапі; ключовий механізм запобігання перевантаженню команди й виявлення вузьких місць.
4. **Управління потоком** (Manage Flow) — моніторинг і оптимізація руху завдань дошкою для забезпечення плавного, передбачуваного потоку.
5. **Явні політики процесу** — чіткі, задокументовані правила («що означає готовність до переходу в наступну колонку»).
6. **Впровадження циклів зворотного зв'язку** — регулярні наради (наприклад, «Kanban-зустрічі», replenishment meeting, delivery planning) для перегляду й адаптації процесу.
7. **Спільне вдосконалення через експерименти** (на основі наукового методу) з використання моделей і метрик для прийняття рішень щодо змін.

## Порівняльна характеристика Scrum і Kanban

| Ознака                      | Scrum                                                  | Kanban                                                                                                          |
|-----------------------------|--------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------|
| Ритм роботи                 | Фіксовані ітерації (спринти)                           | Безперервний потік                                                                                              |
| Ролі                        | Формально визначені                                    | Не регламентовані фреймворком                                                                                   |
| Зміни в ході циклу          | Небажані в межах спринту                               | Допустимі в будь-який момент (з урахуванням WIP-лімітів)                                                        |
| Основний механізм обмеження | Тривалість спринту                                     | WIP-ліміти                                                                                                      |
| Ключова метрика             | обсяг роботи, який команда стабільно виконує за спринт | час від початку активної роботи над завданням до його завершення; кількість завдань, завершених за одиницю часу |
| Область застосування        | Проектні команди з чітким продуктовим фокусом          | Підтримка, сервісні команди, потоки зі змінним навантаженням                                                    |

## 4. Цифрові системи управління проєктами в умовах застосування гнучких підходів

| Клас систем                                            | Призначення                                              | Приклади                                                       |
|--------------------------------------------------------|----------------------------------------------------------|----------------------------------------------------------------|
| Спеціалізовані Agile/Scrum-платформи                   | Повний цикл беклогу, спринтів, дошок, звітності          | Jira (Atlassian), Azure Boards (Microsoft), YouTrack           |
| Візуальні Kanban-дошки                                 | Легка візуалізація потоку робіт                          | Trello, Kanbanize, LeanKit                                     |
| Універсальні системи управління роботою (Work OS)      | Гнучке налаштування процесів різного типу                | Monday.com, ClickUp, Asana, Wrike, Worksection                 |
| Інструменти для розподілених ретроспектив і оцінювання | Підтримка окремих Agile-церемоній                        | Miro/FigJam (віртуальні дошки), Planning Poker Online, Retrium |
| Інтегровані DevOps-платформи                           | Поєднання управління проєктом з розробкою й розгортанням | GitLab, GitHub Projects, Azure DevOps                          |

## Jira від Atlassian. URL: <https://www.atlassian.com/>

Джира

Пошук

+ Створити

Оновлення

Простори

**Житомирська політехніка**

Короткий зміст **Дошка** Список Календар Хронологія Форми Docs Більше 5 +

Пошукова дошка Фільтр Група

Зробити 0 У процесі 0 Готово 0

Немає робочих елементів

Створіть робочий елемент, щоб розпочати. Робота з'явиться тут.

+ Створити

Для тебе

Нещодавні

Позначено зірочкою

Програми

Плани

Простори

Нещодавні

**Житомирська політехніка**

Більше просторів

Рекомендовано

Збирати запити

Спробуйте

Фільтри

Панелі інструментів

Teams

Goals

Projects

Налаштувати бічну панель

# Kanban дошка в Trello

The screenshot shows a Trello Kanban board titled "Мобільний застосунок Житомирська політехніка". The board is organized into five columns: Backlog (8 items), To Do (3 items), In Progress (2 items), Review (1 item), and Done (1 item). Each column contains task cards with descriptions and a "Додати картку" (Add card) button at the bottom. A tooltip "Редагувати картку" (Edit card) is visible over the 12th item in the Backlog column. The background of the board is a city skyline at night. The top navigation bar includes a search bar, a "Створити" (Create) button, and user avatars. The bottom navigation bar includes buttons for "Вхідні" (Inbox), "Планувальник" (Calendar), "Дошка" (Board), and "Переключити дошки" (Switch boards).

**Мобільний застосунок Житомирська політехніка**

**Backlog** (8 items)

- 8. Розробити авторизацію користувача
- 9. Розробити профіль студента
- 10. Розробити модуль «Розклад»
- 11. Розробити модуль «Оцінки»
- 12. Додати університетські новини
- 13. Додати push-сповіщення
- + Додати картку

**To Do** (3 items)

- 4. Визначити основні функції застосунку
- 5. Визначити мінімальний набір функцій, необхідний для першої версії застосунку
- 6. Створити структуру застосунку
- + Додати картку

**In Progress** (2 items)

- 3. Створити User Persona (описати типового користувача)
- 7. Створити прототип головного екрана
- + Додати картку

**Review** (1 item)

- 2. Визначити цільову аудиторію
- + Додати картку

**Done** (1 item)

- 1. Проаналізувати студентів
- + Додати картку

**Bottom Navigation:**

- Вхідні
- Планувальник
- Дошка**
- Переключити дошки

Дякую за увагу!