DOI 10.36074/logos-05.09.2025.025

# ВПРОВАДЖЕННЯ AGILE ДЛЯ РОЗРОБКИ НЕЙРОМЕРЕЖЕВИХ СИСТЕМ

**Копач Богдан Васильович[1], Пелещак Роман Михайлович[2], Пелещак Іван Романович[3]**

**1.** аспірант кафедри інформаційних систем та мереж
*Національний університет «Львівська політехніка», УКРАЇНА*
**ORCID ID: 0009-0002-5158-589X**

**2.** д-р.фіз.-мат.наук, професор, професор кафедри інформаційних систем та мереж
*Національний університет «Львівська політехніка», УКРАЇНА*
**ORCID ID: 0000-0002-0536-3252**

**3.** доктор філософії зі спеціальності 124 «Системний аналіз»,
доцент кафедри інформаційних систем та мереж
*Національний університет «Львівська політехніка», УКРАЇНА*
**ORCID ID: 0000-0002-7481-8628**

Штучний інтелект та нейронні мережі останнім часом стрімко впроваджуються в різних галузях – від медицини до фінансів. Однак успішна реалізація таких проєктів залишається складним завданням. Розробка систем на основі нейронних мереж є процесом, який охоплює збір і підготовку даних, експериментальне налаштування моделей та їхню інтеграцію в програмні продукти. Традиційні послідовні методології розробки не завжди ефективні для таких проєктів через невизначеність вимог і результатів проєкту. Тому більшість спеціалістів для реалізації нейромережевих систем використовують гнучкі методології розробки (англ. Agile), які були спеціально створені для середовищ, що є складними, динамічними та можуть мати лише частково визначені функціональні вимоги [1]. Проте безпосереднє перенесення стандартних Agile-практик на сферу машинного навчання породжує нові виклики. Нейромережеві проєкти мають еволюційний характер вимог та експериментальний масштаб: цілі та критерії успіху можуть змінюватися в процесі досліджень, обсяг та якість доступних даних часто зумовлюють напрямок роботи, а кінцевий результат важко прогнозувати заздалегідь.

Гнучкі методології розробки вперше почали використовувати в індустрії програмного забезпечення. Вони базуються на ряді принципів, які були сформульовані у 2001 році та отримали назву «Agile Manifesto» [2]. У них зазначено, що готовність до змін, тісна співпраця із замовником, поступове створення робочого продукту та взаємодія між членами команди є критично важливими для досягнення успіху проєкту. Найбільш розповсюджені фреймворками Agile – це Scrum [5] та Kanban [1]. Окрім адаптації традиційних Agile методів до AI-проєктів, на практиці все частіше використовують спеціалізовані методології для керування проєктами з обробки даних або машинного навчання. У роботі [4] автори пропонують новий підхід — Data Driven Scrum (DDS), який забезпечує більшу гнучкість і краще відповідає потребам команд, що працюють у сфері науки про дані. DDS виник як відповідь на обмеження класичного Scrum та фокусує увагу не на фіксованих часових спринтах, а на «ітераціях на основі можливостей», які тривають до логічного завершення аналітичного експерименту. У статті [3] було представлено методологію MAISTRO — гнучкий та інтегративний підхід до управління проєктами з розробки систем штучного інтелекту. На відміну від традиційних фреймворків, MAISTRO об'єднує практики Scrum і Kanban з етичними принципами, управлінською прозорістю та стратегічною орієнтацією на бізнес-результати.

Аналізуючи останні дослідження та методології, бачимо, що попри наявність структурованих підходів до розробки систем на основі штучного інтелекту, все ще бракує універсальних практичних рекомендацій для ефективного впровадження Agile-принципів у контекст машинного навчання. Успішність таких проєктів значною мірою залежить від здатності команд адаптуватися до динамічних вимог, які є типовими для сучасних проєктів. Відтак існує потреба в розробці набору рекомендацій, інструментів та шаблонів, які б дозволили інтегрувати Agile в проєкти машинного навчання з урахуванням їхньої специфіки та складності.

Розробка систем на основі нейронних мереж за методологіями Agile передбачає адаптацію стандартного процесу інженерії програмного забезпечення до їхніх особливостей. Створення будь-якої нейронної мережі складається із трьох основних етапів: проєктування архітектури, навчання та розгортання. Кожен із цих етапів має свої специфічні виклики, які впливають на підхід до організації роботи в рамках гнучких методологій. Саме особливості кожного із цих етапів, а не загальноприйняті шаблони класичного життєвого циклу програмного забезпечення визначають процес впровадження Agile.

На етапі проєктування команда уточнює цілі проєкту, формалізує бізнес-вимоги та визначає показники, за якими оцінюватимуть успішність моделі. Паралельно збираються й аналізуються потреби замовника та кінцевих користувачів і перевіряється наявність релевантних даних. Тут вирішують, які саме дані потрібні, де їх отримати та як перетворити у формат, придатний для навчання. Одночасно закладається початкова архітектура нейронної мережі або обираються алгоритми, що найкраще відповідають поставленому завданню. Ключовим результатом цієї фази є план проєкту з розподілом ролей, попередньою оцінкою обсягу робіт і чітко визначеними метриками. Для впровадження Agine на цьому етапі необхідно створити попередній набір гіпотез щодо архітектурних рішень, які команда перевірятиме протягом ітерацій розробки. Для складніших завдань слід провести архітектурні «спайки»: короткі дослідні серії, покликані швидко зняти технічні ризики й уточнити вимоги до обчислювальних ресурсів. Команда повинна одразу визначити мінімальний прототип, який не лише дозволить перевірити концепції проєкту, а й допоможе отримати зворотний зв'язок від замовника. Паралельно необхідно підготувати шаблони для автоматизованих експериментів, які в подальшому дозволять скоротити час між ідеєю й перевіркою.

Упродовж фази навчання відбувається ітеративний процес проведення експериментів над моделлю. Спеціалісти із машинного навчання здійснюють детальне вивчення зібраних даних, їх фільтрацію та підготовку для навчання нейронної мережі. Далі команда розробляє прототипи моделей, випробовуючи різні архітектури нейронних мереж або алгоритми машинного навчання. Кожна ітерація розглядається як окремий експеримент: висувається гіпотеза (наприклад, що додавання певних характеристик даних або зміна гіперпараметрів навчання покращить точність), проводиться навчання моделі та оцінюються результати із використанням метрик, які дозволяють кількісно оцінити вихідні дані. Для впровадження Agile методології на цьому етапі слід встановити короткі цикли експериментів. Замість тривалого монолітного створення моделі команді слід зосередитися на отриманні працездатного прототипу, навіть якщо його функціонал буде обмежений або модель часто генеруватиме помилки. Далі прототип поступово удосконалюється: додаються нові вхідні дані, покращується архітектура нейронної мережі. Фаза навчання часто потребує значних обчислювальних ресурсів і часу (деякі моделі тренуються годинами або днями), тому планування ітерацій розробки необхідно здійснювати не за класичним підходом (фіксований набір завдань на 2-3 тижні), а орієнтуватися на результат: спринт вважатиметься завершеним, коли отримано нову версію нейронної моделі або звіт із кількісними оцінками її якості.

Коли модель продемонструвала прийнятну ефективність у лабораторних умовах, її інтегрують у продуктову систему. Цей етап передбачає різні інженерні завдання: оптимізацію коду моделі для кінцевого середовища, розробку прикладних програмних або користувацьких інтерфейсів для використання моделі користувачами, налаштування інфраструктури серверів для розгортання та забезпечення механізмів моніторингу помилок та продуктивності в реальному часі. На цьому етапі Agile методологію застосовувати найпростіше, оскільки це практично не відрізнятиметься від звичайних IT-проєктів. Команда може використовувати Scrum та працювати короткими спринтами, поступово додаючи необхідну функціональність, наприклад, реалізувати модулі опрацювання вхідних даних або сервіси, що викликатимуть модель. Основною відмінністю в порівнянні із розробкою «звичайного» програмного забезпечення є те, що необхідно контролювати не лише функціональні показники системи, а й метрики якості моделі. Одними із найбільш критичних параметрів є точність та швидкодія нейронної мережі після розгортання. У випадку, якщо вони є незадовільними, команді необхідно розробити алгоритм, що дозволить досягнути бажаних показників. Після розгортання певної версії моделі її переводять у режим моніторингу та підтримки. Використання Agile-підходів на цій стадії гарантує, що оновлення моделі та системи відбуватимуться регулярно з мінімумом ризиків для бізнесу.

**Висновки.** Використання гнучких методологій розробки в процесі створення нейромережевих систем є перспективним підходом, який здатний суттєво підвищити ефективність і результативність таких проєктів. Agile може використовуватися в умовах невизначеності, сприяє швидкому отриманню зворотного зв'язку від замовника, покращує міждисциплінарну взаємодію й дозволяє тримати фокус всієї команди на процесі створення програмного продукту у сфері штучного інтелекту.

## СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ:

[1] *Junior, M., & Godinho Filho, M. (2010). Variations of the kanban system: Literature review and classification. International Journal of Production Economics, 125(1), 13–21. https://doi.org/10.1016/j.ijpe.2010.01.009.*
[2] *Manifesto for Agile Software Development. https://agilemanifesto.org.*
[3] *Petrin, N. S. M., Néto, J. C., & Mariano, H. C. (2025). MAISTRO: Towards an agile methodology for AI system development projects. Applied Sciences, 15(5). https://doi.org/10.3390/app15052628.*
[4] *Saltz, J. S., Sutherland, A., & Hotz, N. (2022). Achieving lean data science agility via data driven Scrum. In Proceedings of the 55th Hawaii International Conference on System Sciences, 1–10. https://hdl.handle.net/10125/80218.*
[5] *Schwaber, K. (1997). Scrum development process. In Business Object Design and Implementation. Springer, London, 117–134. https://doi.org/10.1007/978-1-4471-0947-1_11.*