ĐỔI MỚI DẠY HỌC CÔNG NGHỆ PHẦN MỀM TRONG BỐI CẢNH TRÍ TUỆ NHÂN TẠO TẠO SINH: TỪ NĂNG LỰC LẬP TRÌNH ĐẾN NĂNG LỰC PHÁT TRIỂN VÀ KIỂM CHỨNG PHẦN MỀM

Nguyễn Thị Lan Anh
Khoa Công nghệ Thông tin, Trường Đại học Công nghệ Kỹ thuật Vinh
Email: anhntl.skv@gmail.com

Tóm tắt: *Sự phát triển của trí tuệ nhân tạo tạo sinh đang làm thay đổi đáng kể hoạt động phát triển phần mềm, đặc biệt ở khả năng sinh mã, giải thích chương trình, phát hiện lỗi và đề xuất phương án xử lý. Điều này đặt ra yêu cầu đổi mới cách tiếp cận trong dạy học Công nghệ phần mềm, bởi năng lực của người học không nên chỉ được thể hiện qua khả năng viết mã mà còn phải bao gồm phân tích yêu cầu, thiết kế giải pháp, kiểm chứng, kiểm thử, đánh giá và bảo trì phần mềm. Bài viết phân tích đặc điểm của học phần Công nghệ phần mềm trong mối quan hệ với sự phát triển của trí tuệ nhân tạo tạo sinh, từ đó đề xuất định hướng tổ chức dạy học theo chuỗi phân tích – thiết kế – lập trình có hỗ trợ AI – kiểm chứng – kiểm thử – bảo trì. Trong cách tiếp cận này, AI được xem là công cụ hỗ trợ có kiểm soát, còn sinh viên vẫn giữ vai trò chủ thể trong việc lựa chọn giải pháp, đánh giá kết quả và chịu trách nhiệm đối với sản phẩm phần mềm. Bài viết mang tính định hướng, không sử dụng dữ liệu khảo sát hoặc kết quả thực nghiệm.*

Từ khóa: *Công nghệ phần mềm, trí tuệ nhân tạo tạo sinh, dạy học, lập trình, kiểm chứng phần mềm, kiểm thử phần mềm.*

INNOVATING SOFTWARE ENGINEERING EDUCATION IN THE CONTEXT OF GENERATIVE ARTIFICIAL INTELLIGENCE: FROM PROGRAMMING SKILLS TO SOFTWARE DEVELOPMENT AND VERIFICATION COMPETENCIES

Abstract: *The development of generative artificial intelligence (AI) is significantly transforming software development practices, particularly through its capabilities in code generation, program explanation, error detection, and solution suggestion. This development calls for a renewed approach to Software Engineering education, as learners' competencies should not be demonstrated solely through their ability to write code but should also encompass requirements analysis, solution design, verification, testing, evaluation, and software maintenance. This paper analyzes the characteristics of Software Engineering as a course in relation to the development of generative AI and proposes an instructional approach based on the sequence of analysis – design – AI-assisted programming – verification – testing – maintenance. In this approach, AI is considered a controlled support tool, while students remain the primary agents responsible for selecting solutions, evaluating results, and taking responsibility for their software products. The paper presents an instructional orientation and does not use survey data or empirical results.*

Keywords: *Software Engineering, generative artificial intelligence, education, programming, software verification, software testing.*

Nhận bài: 18/09/2026 *Phản biện: 18/10/2026* *Duyệt đăng: 22/10/2026*

I. ĐẶT VẤN ĐỀ

Công nghệ phần mềm là học phần nền tảng của đào tạo Công nghệ thông tin, bao quát toàn bộ quá trình phát triển và duy trì phần mềm, từ phân tích, thiết kế đến lập trình, kiểm thử và bảo trì. Vì vậy, mục tiêu của học phần không chỉ là tạo ra chương trình chạy được mà còn hình thành khả năng tiếp cận có hệ thống đối với quá trình xây dựng sản phẩm phần mềm (Pressman & Maxim, 2020; Sommerville, 2016).

Trí tuệ nhân tạo tạo sinh đang hỗ trợ ngày càng nhiều công việc như giải thích yêu cầu, đề xuất thuật toán, sinh mã, chuyển đổi mã, phát hiện lỗi và tạo trường hợp kiểm thử. GitHub Copilot là một ví dụ về công cụ hỗ trợ lập trình dựa trên ngữ cảnh (GitHub, 2024). Các nghiên cứu về mô hình ngôn ngữ lớn trong công nghệ phần mềm cũng cho thấy AI có thể tham gia nhiều nhiệm vụ trong vòng đời phát triển, nhưng kết quả vẫn cần con người kiểm tra và đánh giá (Fan et al., 2023; Hou et al., 2024).

Trong bối cảnh đó, nếu đánh giá chủ yếu bằng khả năng viết một đoạn mã hoặc hoàn thành một chức năng, năng lực người học có thể không được phản ánh đầy đủ. Ngược lại, nếu chú trọng hiểu bài toán, phân tích yêu cầu, thiết kế, kiểm chứng mã, kiểm thử, đánh giá và bảo trì, sinh viên được đặt vào vai trò của người phát triển phần mềm. Do đó, bài viết đề xuất chuyển trọng tâm từ "năng lực lập trình" sang "năng lực phát triển và kiểm chứng phần mềm", trong đó AI hỗ trợ nhưng không thay thế trách nhiệm chuyên môn của sinh viên.

II. YÊU CẦU ĐỔI MỚI DẠY HỌC CÔNG NGHỆ PHẦN MỀM TRONG BỐI CẢNH AI TẠO SINH

Một sản phẩm phần mềm không bắt đầu từ viết mã mà từ xác định vấn đề, làm rõ yêu cầu, phân tích đối tượng và chức năng, lựa chọn kiến trúc, thiết kế giao diện và dữ liệu, rồi mới lập trình,

kiểm thử và bảo trì (Pressman & Maxim, 2020; Sommerville, 2016). Bản chất của Công nghệ phần mềm vì vậy không nằm ở số dòng mã mà ở khả năng sử dụng phương pháp và công cụ phù hợp để giải quyết bài toán.

AI có thể tạo mã từ yêu cầu bằng ngôn ngữ tự nhiên, nhưng mã được sinh ra chưa đồng nghĩa với một giải pháp đúng, an toàn, hiệu quả và có khả năng bảo trì. Mã có thể chạy nhưng vẫn chứa lỗi logic, bỏ sót trường hợp biên, không phù hợp kiến trúc hoặc tạo rủi ro bảo mật. Vì vậy, sinh viên cần đủ kiến thức nền tảng để đọc hiểu và đánh giá đầu ra của AI.

Mục tiêu dạy học có thể tổ chức thành chuỗi năng lực: (1) hiểu và phân tích bài toán, xác định yêu cầu chức năng và phi chức năng; (2) thiết kế cấu trúc hệ thống, dữ liệu, giao diện và các thành phần; (3) lập trình với sự hỗ trợ của AI nhưng phải đọc hiểu mã; (4) kiểm chứng và kiểm thử để xác định mức độ đáp ứng yêu cầu; (5) bảo trì, sửa lỗi và đánh giá tác động của thay đổi. Như vậy, sinh viên vẫn cần học lập trình, nhưng lập trình được đặt trong hệ thống năng lực rộng hơn: phân tích – thiết kế – lập trình – kiểm chứng – kiểm thử – bảo trì.

III. ĐỀ XUẤT ĐỔI MỚI CÁCH TỔ CHỨC DẠY HỌC CÔNG NGHỆ PHẦN MỀM

Trên cơ sở cấu trúc học phần gồm 28 tiết, có thể tổ chức nội dung theo một chuỗi thống nhất thay vì xem ba chương như những phần kiến thức độc lập. Chương 1 cung cấp nền tảng về phần mềm và quá trình phát triển phần mềm; Chương 2 tập trung vào phân tích và thiết kế; Chương 3 đề cập đến lập trình, kiểm thử và bảo trì. Ba chương này có thể được kết nối thành một quy trình nhận thức và thực hành thống nhất: hiểu bài toán → phân tích → thiết kế → lập trình → kiểm chứng → kiểm thử → bảo trì. Cách tổ chức này giúp sinh viên nhận thức rõ mối quan hệ giữa các hoạt động, đồng thời tạo điều kiện đưa AI vào đúng vị trí hỗ trợ thay vì biến AI thành nội dung học tập trung tâm.

Ở Chương 1 – Tổng quan về Công nghệ phần mềm, mục tiêu không chỉ dừng ở việc cung cấp định nghĩa và lịch sử phát triển mà cần giúp sinh viên hình thành nhận thức về bản chất của một sản phẩm phần mềm và vòng đời phát triển phần mềm. Giảng viên có thể đưa ra một bài toán đơn giản và yêu cầu sinh viên xác định phần mềm cần xây dựng, đối tượng sử dụng, phạm vi chức năng và những vấn đề cần giải quyết. AI có thể được sử dụng để hỗ trợ tìm hiểu các khái niệm hoặc gợi ý những câu hỏi cần đặt ra đối với bài toán. Tuy nhiên, sinh viên cần tự lựa chọn thông tin phù hợp và giải thích được lý do xác định phạm vi của hệ thống. Qua đó, ngay từ đầu học phần, sinh viên được hình thành thói quen không xem kết quả do AI cung cấp là đáp án mặc nhiên đúng.

Chương 2 – Phân tích và thiết kế hệ thống phần mềm cần được xem là giai đoạn hình thành năng lực có ý nghĩa đặc biệt trong bối cảnh AI có thể sinh mã. Khi công cụ có khả năng tạo ra chương trình từ yêu cầu bằng ngôn ngữ tự nhiên, khả năng xác định đúng yêu cầu và thiết kế giải pháp càng trở nên quan trọng. Sinh viên cần thực hiện các hoạt động phân tích hệ thống, xác định yêu cầu, mô hình hóa hệ thống theo hướng cấu trúc hoặc hướng đối tượng, thiết kế cơ sở dữ liệu, giao diện và các thành phần của hệ thống. AI có thể hỗ trợ giải thích ký pháp, đưa ra một số phương án thiết kế hoặc phản biện một thiết kế đã có. Tuy nhiên, sinh viên phải là người lựa chọn phương án cuối cùng và giải thích được mối quan hệ giữa yêu cầu, mô hình và thiết kế. Việc đánh giá ở giai đoạn này vì vậy không nên chỉ dựa trên việc sơ đồ có đúng hình thức hay không mà cần chú trọng khả năng giải thích và bảo vệ quyết định thiết kế.

Đối với Chương 3 – Lập trình, kiểm thử và bảo trì phần mềm, AI có thể được khai thác rõ hơn nhưng cần thiết lập nguyên tắc sử dụng có kiểm soát. Sinh viên có thể sử dụng AI để tạo một đoạn mã ban đầu, giải thích một đoạn mã chưa hiểu, tìm nguyên nhân của lỗi hoặc đề xuất trường hợp kiểm thử. Sau đó, sinh viên phải thực hiện quá trình kiểm tra bằng cách chạy chương trình, đọc mã nguồn, đối chiếu với yêu cầu và kiểm thử các trường hợp phù hợp. Một hoạt động học tập có thể được tổ chức theo chu trình: Yêu cầu → Sinh viên thiết kế → AI hỗ trợ mã → Sinh viên đọc hiểu → Kiểm thử → Phát hiện lỗi → Sửa chữa → Giải thích kết quả.

Chu trình này làm thay đổi đáng kể vai trò của hoạt động lập trình. Sinh viên không chỉ được yêu cầu "viết một chương trình chạy được" mà phải chứng minh rằng chương trình đáp ứng yêu cầu và có thể được kiểm chứng. Chẳng hạn, với một chức năng đăng nhập, thay vì chỉ yêu cầu sinh viên tạo giao diện và mã xử lý đăng nhập, giảng viên có thể yêu cầu sinh viên xác định các trường hợp đầu vào hợp lệ và không hợp lệ, kiểm tra xử lý mật khẩu, kiểm tra thông báo lỗi và phân tích các đoạn mã do AI đề xuất. Khi đó, kiến thức lập trình vẫn được sử dụng nhưng được đặt trong một nhiệm vụ có yêu cầu cao hơn về kiểm chứng và chất lượng.

Một thay đổi quan trọng khác là đánh giá sinh viên. Trong môi trường có AI tạo sinh, nếu bài kiểm tra chủ yếu yêu cầu viết một đoạn mã ngắn từ đầu, việc đánh giá có thể chưa phản ánh đầy đủ năng lực phát triển phần mềm của người học. Có thể mở rộng đánh giá theo các thành phần: phân tích yêu cầu, thiết kế, lập trình, kiểm chứng, kiểm thử, bảo trì và khả năng giải thích. Đối với sản phẩm có sử dụng AI, sinh viên có thể phải ghi lại công cụ đã sử dụng, mục đích sử dụng, phần mã hoặc nội dung được AI hỗ trợ, những thay đổi đã thực hiện và cách kiểm tra kết quả. Cách làm này không nhằm cấm AI mà hướng sinh viên đến việc sử dụng AI có trách nhiệm và có khả năng kiểm soát.

Bên cạnh đó, vai trò của giảng viên cũng cần được điều chỉnh. Khi AI có thể cung cấp nhanh nhiều câu trả lời và đoạn mã, giảng viên không nhất thiết phải dành phần lớn thời gian để hướng dẫn từng thao tác lập trình cơ bản. Thay vào đó, giảng viên có thể tập trung nhiều hơn vào việc thiết kế tình huống, đặt vấn đề, yêu cầu sinh viên giải thích lựa chọn, kiểm tra sản phẩm, chỉ ra những trường hợp mà AI đưa ra kết quả chưa phù hợp và hướng dẫn sinh viên tự kiểm chứng. Như vậy, AI đảm nhận một phần hoạt động hỗ trợ kỹ thuật, trong khi giảng viên giữ vai trò thiết kế học tập, định hướng chuyên môn và phản hồi; sinh viên vẫn là chủ thể chịu trách nhiệm về kết quả học tập và sản phẩm phần mềm.

Một nguyên tắc cần được duy trì xuyên suốt học phần là "AI có thể hỗ trợ tạo ra kết quả nhưng không thay thế trách nhiệm kiểm chứng kết quả". Sinh viên có thể sử dụng AI để tăng tốc một số công việc nhưng phải hiểu rằng đầu ra của AI có thể không chính xác, không phù hợp với ngữ cảnh hoặc không đáp ứng đầy đủ yêu cầu. Vì vậy, mọi nội dung do AI hỗ trợ cần được đặt trong một chu trình kiểm tra gồm đối chiếu yêu cầu, kiểm tra mã, thực thi, kiểm thử và đánh giá. Đây là điểm có thể tạo nên sự khác biệt giữa việc "sử dụng AI để làm bài" và "học Công nghệ phần mềm trong môi trường có AI".

IV. KẾT LUẬN

Trí tuệ nhân tạo tạo sinh đang làm thay đổi hoạt động phát triển phần mềm, đặc biệt với khả năng sinh mã, qua đó đặt ra yêu cầu mới đối với đào tạo Công nghệ thông tin. Đổi mới dạy học Công nghệ phần mềm vì vậy không chỉ là đưa AI vào thực hành lập trình mà cần mở rộng trọng tâm sang các năng lực phân tích yêu cầu, thiết kế, đọc hiểu và kiểm chứng mã nguồn, kiểm thử, bảo trì và giải thích quyết định kỹ thuật.

Trên cơ sở cấu trúc học phần, bài viết đề xuất tổ chức nội dung theo chuỗi phân tích – thiết kế – lập trình có hỗ trợ AI – kiểm chứng – kiểm thử – bảo trì. AI được sử dụng như công cụ hỗ trợ, còn sinh viên giữ vai trò chủ thể trong xác định yêu cầu, lựa chọn giải pháp, đánh giá đầu ra và chịu trách nhiệm về sản phẩm. Cách tiếp cận này vừa duy trì vai trò của kiến thức lập trình, vừa mở rộng mục tiêu đào tạo theo yêu cầu phát triển phần mềm trong bối cảnh mới. Đây là định hướng tổ chức dạy học dựa trên đặc điểm học phần và sự thay đổi công nghệ, chưa khẳng định hiệu quả qua thực nghiệm. Các nghiên cứu tiếp theo có thể triển khai mô hình ở các lớp học cụ thể để hoàn thiện nội dung, phương thức đánh giá và đánh giá tác động đối với năng lực phát triển phần mềm của sinh viên.

TÀI LIỆU THAM KHẢO

Fan, A., Gokkaya, B., Harman, M., Lyubarskiy, M., Sengupta, S., Yoo, S., & Zhang, J. M. (2023). Large language models for software engineering: Survey and open problems. In 2023 IEEE/ACM International Conference on Software Engineering: Future of Software Engineering (ICSE-FoSE) (pp. 31–53). IEEE. https://doi.org/10.1109/ICSE-FoSE59343.2023.00008

GitHub. (2024). GitHub Copilot documentation. https://docs.github.com/en/copilot

Hou, X., Zhao, Y., Liu, Y., Yang, Z., Wang, K., Li, L., Luo, X., Lo, D., Grundy, J., & Wang, H. (2024). Large language models for software engineering: A systematic literature review. ACM Transactions on Software Engineering and Methodology, 33(8), Article 220. https://doi.org/10.1145/3695988

Nguyễn Việt Hà. (2015). Công nghệ phần mềm. Nhà xuất bản Đại học Quốc gia Hà Nội.

Pressman, R. S., & Maxim, B. R. (2020). Software engineering: A practitioner's approach (9th ed.). McGraw-Hill.

Sommerville, I. (2016). Software engineering (10th ed.). Pearson.