Journal of KIIT. Vol. 24, No. 8, pp. 231-241, Aug. 31, 2026. pISSN 1598-8619, eISSN 2093-7571 
http://dx.doi.org/10.14801/jkiit.2026.24.8.231

# AI 코딩 도구 활용에서 AI 의존, 지각된 생산성 및 기술부채 위험 간 구조적 관계 분석

정성채*, 안현철**

# A Structural Relationship Analysis among AI Dependence, Perceived Productivity, and Technical Debt Risk in AI Coding Tool Use

Sungche Jung*, Hyunchul Ahn**


## 요 약

생성형 AI 기반 코딩 도구는 코드 작성, 오류 수정, 문서화 등 개발 활동을 지원하며 소프트웨어 개발 방식에 변화를 가져오고 있다. 그러나 이러한 도구는 생산성을 높이는 동시에 AI 산출물에 대한 과도한 의존과 기술부채 위험을 초래할 수 있다. 본 연구는 AI 코딩 도구 활용 경험이 있는 개발 관련 종사자 200명을 대상으로 AI 신뢰 성향, 검증 성향, AI 사용 빈도가 AI 의존 경향에 미치는 영향과 AI 의존 경향이 지각된 생산성 및 기술부채 위험에 미치는 영향을 PLS-SEM으로 분석하였다. 분석 결과, AI 신뢰 성향은 AI 의존 경향에 유의한 영향을 미치지 않았으나, 검증 성향은 AI 의존 경향을 낮추고 AI 사용 빈도는 AI 의존 경향을 높였다. 또한 AI 의존 경향은 지각된 생산성과 기술부채 위험을 모두 높였다. 본 연구는 AI 코딩 도구 활용이 생산성 향상과 기술부채 위험 증가라는 양면적 결과를 가질 수 있음을 시사한다.

## Abstract

AI coding tools based on generative AI are changing software development by supporting code generation, debugging, documentation, and related tasks. This study examines how AI trust propensity, verification orientation, and AI use frequency affect AI dependence and how AI dependence influences perceived productivity and technical debt risk. Using 200 responses from development-related workers with experience using AI coding tools, the research model was tested with partial least squares structural equation modeling (PLS-SEM). The results show that AI trust propensity does not significantly affect AI dependence, whereas verification orientation negatively affects AI dependence and AI use frequency positively affects AI dependence. AI dependence has positive effects on both perceived productivity and technical debt risk. These findings suggest that AI coding tool use can generate ambivalent outcomes: productivity gains and increased technical debt risk.



Keywords

AI coding tools, AI dependence, verification orientation, perceived productivity, technical debt risk, PLS-SEM



* 국민대학교 비즈니스IT대학원 박사과정
- ORCID: https://orcid.org/0009-0004-3493-5063
** 국민대학교 비즈니스IT대학원 교수(교신저자)
- ORCID: https://orcid.org/0000-0003-1867-6423



· Received: May 06, 2026, Revised: May 22, 2026, Accepted: May 25, 2026


· Corresponding Author: Hyunchul Ahn
Graduate School of Business IT, Kookmin University
Seoul, Korea
Tel.: +82-2-910-4577, Email: hcahn@kookmin.ac.kr

# Ⅰ. 서 론

생성형 인공지능 기술의 확산은 소프트웨어 개발 방식에 중요한 변화를 가져오고 있다. 최근 개발자들은 ChatGPT, GitHub Copilot, Cursor, Claude 등 다양한 AI 코딩 도구를 활용하여 코드 작성, 오류 수정, 리팩토링, 테스트 코드 생성, 문서화, API 사용 예시 탐색 등 여러 개발 활동을 수행하고 있다. AI 코딩 도구는 자연어 기반의 질의와 응답을 통해 개발자의 문제 해결을 지원하며, 기존의 코드 자동완성 도구를 넘어 개발 과정 전반에 관여하는 협업적 도구로 발전하고 있다[1][2].

AI 코딩 도구의 확산은 개발 생산성 향상에 대한 기대를 높이고 있다. 개발자는 AI의 도움을 통해 반복적인 코드 작성 시간을 줄이고, 오류 해결을 위한 탐색 시간을 단축하며, 새로운 라이브러리나 프레임워크 사용에 필요한 정보를 빠르게 확보할 수 있다. 특히 생성형 AI 기반 도구는 코드 조각뿐 아니라 설명, 대안, 테스트 코드, 설계 아이디어까지 함께 제안할 수 있기 때문에 개발자의 업무 효율을 높이는 수단으로 주목받고 있다[3][4].

그러나 AI 코딩 도구 활용이 항상 긍정적 결과만을 가져오는 것은 아니다. AI가 생성한 코드는 문법적으로는 작동하더라도 프로젝트의 구조, 보안 요구사항, 유지보수성, 확장성, 조직의 개발 표준을 충분히 반영하지 못할 수 있다. 또한 개발자가 AI 산출물을 충분히 검증하지 않고 수용할 경우 코드의 내부 논리와 설계 의도에 대한 이해가 약화될 수 있으며, 이는 장기적으로 기술부채 위험을 증가시킬 수 있다[5][6]. 따라서 AI 코딩 도구 활용은 개발 생산성을 높이는 동시에 코드 품질 및 유지보수성과 관련된 새로운 위험을 수반할 수 있다.

본 연구는 AI 코딩 도구 활용에서 개발자의 AI 의존 경향이 지각된 생산성과 기술부채 위험으로 연결되는 구조적 관계를 검증하고자 한다. 이를 위해 복수의 잠재변수와 매개효과를 동시에 분석할 수 있는 PLS-SEM(Partial Least Squares Structural Equation Modeling)을 적용하였다. PLS-SEM은 탐색적·예측지향적 연구모형에서 잠재변수 간 경로관계와 측정모형의 신뢰도 및 타당도를 함께 검토할 수 있다는 장점이 있다.

기존 연구는 AI 코딩 도구의 생산성 효과, 코드 생성 성능, 인간-AI 협업 가능성에 주목해 왔다[3][5]. 그러나 개발자의 개인적 성향과 사용 경험이 AI 의존 경향을 어떻게 형성하는지, 그리고 AI 의존 경향이 생산성과 기술부채 위험이라는 상반된 결과에 어떠한 영향을 미치는지를 함께 분석한 연구는 충분하지 않다. 본 연구는 AI 의존 경향을 중심으로 지각된 생산성과 기술부채 위험을 동시에 분석한다는 점에서 기존 연구와 차별성을 가진다.

본 연구의 기여점은 다음과 같다.

첫째, AI 코딩 도구 활용에서 개발자의 AI 의존 경향을 중심 변수로 설정하였다.

둘째, AI 의존 경향이 지각된 생산성과 기술부채 위험이라는 양면적 결과로 연결되는 구조를 검증하였다.

셋째, 검증 성향과 AI 사용 빈도가 AI 의존 경향에 미치는 상반된 영향을 실증적으로 제시하였다.

본 논문의 구성은 다음과 같다. Ⅱ장에서는 AI 코딩 도구, AI 신뢰 성향, 검증 성향, AI 사용 빈도, AI 의존 경향, 지각된 생산성 및 기술부채 위험에 관한 이론적 배경을 검토하고 연구가설을 제시한다. Ⅲ장에서는 연구모형, 변수의 조작적 정의, 자료수집 및 분석방법을 설명한다. Ⅳ장에서는 PLS-SEM을 활용한 측정모형과 구조모형 분석결과를 제시한다. 마지막으로 Ⅴ장에서는 연구결과를 요약하고 학술적·실무적 시사점, 연구의 한계 및 향후 연구 방향을 제시한다.

# Ⅱ. 이론적 배경 및 연구가설

## 2.1 AI 코딩 도구와 소프트웨어 개발

AI 코딩 도구는 생성형 인공지능과 대규모 언어모델을 기반으로 개발자의 코딩 활동을 지원하는 도구를 의미한다. 기존의 자동완성 도구가 주로 문법적 보완이나 코드 조각 추천에 초점을 두었다면, 최근의 AI 코딩 도구는 자연어 요구사항을 코드로 변환하거나, 오류의 원인을 설명하고, 대안적 구현 방식을 제안하며, 테스트 코드와 문서까지 생성할

수 있다는 점에서 활용 범위가 넓다[1][21].

최근 연구들은 AI 코딩 도구가 단순한 코드 완성 기능을 넘어 소프트웨어 개발 생명주기 전반에서 활용되고 있음을 보여준다. Sergeyuk et al.[1]은 개발자들이 신규 기능 구현, 테스트 작성, 버그 처리, 리팩토링, 자연어 산출물 작성 등 다양한 개발 활동에서 AI assistant를 활용하고 있으며, 특히 개발자가 덜 선호하거나 위임하고 싶어 하는 작업에서 AI 지원 수요가 높다고 보고하였다. 이러한 결과는 AI 코딩 도구가 특정 코딩 단계에 한정된 보조 도구가 아니라 개발 업무 전반의 생산성과 작업 경험을 재구성하는 도구임을 시사한다.

AI 코딩 도구와 개발자의 상호작용 방식도 점차 다양해지고 있다. Barke et al.[5]은 개발자가 Copilot과 상호작용하는 방식을 acceleration mode와 exploration mode로 구분하였다. acceleration mode에서는 개발자가 이미 다음에 무엇을 할지 알고 있으며, AI는 이를 더 빠르게 구현하도록 돕는다. 반면 exploration mode에서는 개발자가 해결 방향을 탐색하기 위해 AI를 활용하며, 이 경우 더 명시적인 프롬프트 작성과 산출물 검증이 필요하다[5]. Ross et al.[12]은 대화형 프로그래밍 어시스턴트가 코드 생성뿐 아니라 설명, 후속 질문, 문제 해결 방향 탐색 등 더 넓은 협업적 기능을 제공할 수 있음을 보여주었다.

AI 코딩 도구는 개발자의 작업 속도를 높이고 반복적 작업을 줄이는 데 도움을 줄 수 있다. 개발자는 AI를 활용하여 코드 작성의 초기 부담을 낮추고, 익숙하지 않은 기술 스택에 대한 접근성을 높이며, 디버깅과 리팩토링 과정에서 다양한 대안을 탐색할 수 있다. 이러한 특성은 AI 코딩 도구가 개발자의 생산성 향상에 기여할 수 있음을 시사한다[3][4].

반면 AI 코딩 도구의 산출물은 항상 정확하거나 최적화된 결과를 보장하지 않는다. AI가 생성한 코드는 학습 데이터와 프롬프트 맥락에 의존하기 때문에 프로젝트의 세부 요구사항이나 조직의 개발 표준을 충분히 반영하지 못할 수 있다. 또한 개발자가 AI의 제안을 충분히 이해하지 못한 상태에서 코드를 수용하면, 장기적으로 유지보수성 저하와 기술 부채 누적이 발생할 수 있다[6][7]. 따라서 AI 코딩 도구의 활용 효과를 이해하기 위해서는 생산성 효과뿐만 아니라 의존과 검증, 기술부채 위험을 함께 고려할 필요가 있다.

## 2.2 AI 신뢰 성향과 AI 의존 경향

AI 신뢰 성향은 개발자가 AI 코딩 도구의 산출물, 추천, 설명, 문제 해결 능력을 신뢰하는 정도를 의미한다. 정보시스템 및 인간-AI 상호작용 연구에서 신뢰는 기술 수용과 활용을 설명하는 중요한 요인으로 다루어져 왔다. 자동화 시스템 연구에서도 사용자의 신뢰는 자동화 도구의 사용, 오용, 비사용을 설명하는 주요 요인으로 논의되어 왔으며[26], AI 기반 개발 맥락에서도 신뢰는 개발자가 AI에게 어느 정도의 판단 권한을 위임하는지를 설명하는 주요 요인으로 볼 수 있다[5][7].

AI 코딩 도구 활용 맥락에서도 신뢰는 중요한 역할을 할 수 있다. 개발자가 AI 도구의 코드 생성 능력과 문제 해결 능력을 높게 신뢰할수록 AI가 제시한 코드나 설명을 수용할 가능성이 커지고, 이후 유사한 상황에서도 AI의 도움을 반복적으로 요청할 수 있다. 다만 AI 코딩 도구의 신뢰는 단순한 수용 의도와 동일하지 않다. AI 산출물의 정확성, 프로젝트 맥락 이해 수준, 보안과 품질에 대한 우려는 AI 신뢰와 실제 의존 사이의 관계를 약화시킬 수 있다[1][13]. 그럼에도 개발자가 AI를 신뢰할수록 AI를 주요 문제 해결 자원으로 활용할 가능성이 커질 수 있으므로, 본 연구는 AI 신뢰 성향이 AI 의존 경향을 높일 것으로 예상하였다[2][5].

이에 본 연구는 다음과 같은 가설을 설정하였다.

**H1. AI 신뢰 성향은 AI 의존 경향에 정(+)의 영향을 미칠 것이다.**

## 2.3 검증 성향과 AI 의존 경향

검증 성향은 개발자가 AI 코딩 도구의 산출물을 그대로 수용하지 않고, 정확성, 적합성, 코드 품질, 유지보수 가능성 등을 확인하려는 성향을 의미한다. AI 코딩 도구가 빠르게 결과물을 제공하더라도 그

결과가 실제 프로젝트 맥락에서 적절한지는 별도의 검토가 필요하다. 특히 코드의 문법적 오류는 쉽게 발견될 수 있지만, 구조적 설계 문제, 예외 처리 부족, 보안 취약성, 장기적 유지보수성 문제는 즉각적으로 드러나지 않을 수 있다[5][6][14].

검증 성향이 높은 개발자는 AI 산출물을 비판적으로 검토하고, AI가 제시한 코드의 작동 원리와 설계 의도를 확인하려는 경향을 보인다. 선행연구에서도 AI 기반 코딩은 단순한 자동화가 아니라 생성 결과를 빠르게 평가하고 수정하는 반복적 상호작용으로 설명되며, 검증과 테스트는 AI 산출물의 신뢰성을 확보하는 핵심 활동으로 논의된다[5][8]. 또한 현대적 코드 리뷰 연구는 리뷰 활동이 단순한 결함 탐지뿐 아니라 코드 이해, 지식 공유, 대안적 해결책 탐색에 기여한다고 설명한다[15][16]. 따라서 AI 코딩 맥락에서 검증 성향은 개발자가 AI 산출물에 대한 이해와 통제권을 유지하도록 하며, AI에 대한 과도한 의존을 억제하는 역할을 할 수 있다.

이에 본 연구는 다음과 같은 가설을 설정하였다.

**H2. 검증 성향은 AI 의존 경향에 부(-)의 영향을 미칠 것이다.**

### 2.4 AI 사용 빈도와 AI 의존 경향

AI 사용 빈도는 개발자가 AI 코딩 도구를 개발 업무에서 얼마나 자주 활용하는지를 의미한다. AI 사용 빈도는 도구를 얼마나 자주 사용하는지를 나타내는 행동적 사용 수준인 반면, AI 의존 경향은 문제 해결과 판단 과정에서 AI 산출물에 얼마나 의지하는지를 나타내는 심리적·인지적 의존 수준이다. 따라서 두 변수는 관련되어 있으나 동일한 개념은 아니다.

AI 코딩 도구를 자주 사용하는 개발자는 코드 작성, 오류 해결, 예제 탐색, 문서화, 테스트 코드 생성 등 다양한 상황에서 AI를 반복적으로 호출하게 된다. 이러한 반복적 사용은 AI 도구를 보조적 수단을 넘어 개발 업무의 기본적 작업 방식으로 정착시킬 수 있다. 특히 개발자가 스스로 문제를 해결하기 전에 먼저 AI에게 질문하거나, AI가 제시한 방향을 기준으로 개발을 진행하게 될 경우 AI 의존 경향은 강화될 수 있다.

**H3. AI 사용 빈도는 AI 의존 경향에 정(+)의 영향을 미칠 것이다.**

### 2.5 AI 의존 경향과 지각된 생산성

AI 의존 경향은 개발자가 개발 과정에서 AI 코딩 도구의 제안, 코드 생성 결과, 오류 수정 방식, 문제 해결 방향에 의존하는 정도를 의미한다. 본 연구에서 AI 의존 경향은 AI 코딩 도구에 대한 개인적 성향 및 사용 경험이 지각된 결과로 연결되는 과정을 설명하는 매개적 구성개념으로 설정된다. 본 연구의 목적은 AI 의존 경향이 다른 변수의 효과를 강화·약화하는 조건으로 작용하는지를 보기보다, 개발자의 신뢰·검증·사용 경험이 지각된 성과로 이어지는 과정에서 AI 의존 경향이 어떤 경로적 역할을 하는지를 분석하는 데 있다.

AI 의존 경향은 개발자가 개발 과정에서 AI 코딩 도구의 제안, 코드 생성 결과, 오류 수정 방식, 문제 해결 방향에 의존하는 정도를 의미한다. AI 의존 경향은 단순한 사용량과 구분된다. 사용량은 도구를 얼마나 사용하는지를 의미하지만, 의존 경향은 개발자가 문제 해결과 판단 과정에서 AI를 얼마나 필요로 하고 의지하는지를 나타낸다.

AI 의존 경향이 높은 개발자는 반복적 코드 작성, 오류 수정, 정보 탐색, 테스트 코드 생성 등의 과정에서 AI를 적극적으로 활용할 가능성이 높다. 이러한 활용은 개발자의 작업 시간을 단축하고, 문제 해결 속도를 높이며, 개발 업무의 효율성을 향상시킬 수 있다[3][4]. 따라서 AI 의존 경향은 개발자가 지각하는 생산성을 높이는 요인으로 작용할 수 있다.

이에 본 연구는 다음과 같은 가설을 설정하였다.

**H4. AI 의존 경향은 지각된 생산성에 정(+)의 영향을 미칠 것이다.**

### 2.6 AI 의존 경향과 지각된 기술부채 위험

기술부채는 단기적 개발 효율을 위해 선택한 구현 방식이 장기적으로 유지보수 비용, 품질 저하, 확장성 문제를 초래하는 현상을 의미한다[9]. AI 코딩 도구는 빠르게 작동 가능한 코드를 생성할 수 있지만, 생성된 코드가 프로젝트의 전체 구조, 장기적 유지보수성, 보안 요구사항, 예외 처리, 테스트 가능성을 충분히 반영하지 못할 수 있다[6].

AI 의존 경향이 높은 개발자는 AI가 생성한 코드나 설명을 반복적으로 활용하면서 단기적으로는 높은 생산성을 경험할 수 있다. 그러나 AI 산출물에 대한 이해와 검증이 충분하지 않을 경우 코드 품질에 대한 통제력이 낮아지고, 설계 일관성이나 유지보수 가능성이 저하될 수 있다. 보안 관점에서도 AI assistant 접근 권한이 있는 사용자가 더 취약한 코드를 작성하거나 자신이 안전한 코드를 작성했다고 과신할 수 있다는 결과가 보고되었다[13]. 이러한 상황은 AI 의존 경향이 지각된 기술부채 위험을 증가시킬 수 있음을 시사한다.

**H5. AI 의존 경향은 지각된 기술부채 위험에 정(+)의 영향을 미칠 것이다.**

## III. 연구 방법

### 3.1 연구모형 및 변수의 조작적 정의

본 연구는 AI 신뢰 성향, 검증 성향, AI 사용 빈도가 AI 의존 경향에 미치는 영향을 분석하고, AI 의존 경향이 지각된 생산성과 지각된 기술부채 위험에 미치는 영향을 검증하기 위해 그림 1과 같은 연구모형을 설정하였다.

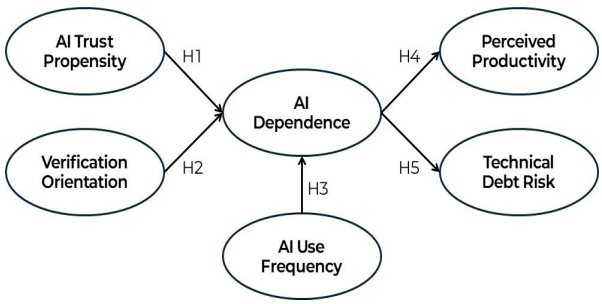


그림 1. 연구모형
Fig. 1. Research model

모든 측정문항은 AI 코딩 도구 활용 경험을 전제로 응답하도록 설계하였으며, 각 문항은 1점(전혀 그렇지 않다)부터 7점(매우 그렇다)까지의 리커트 7점 척도를 사용하여 측정하였다. 본 연구에서 사용된 연구변수의 조작적 정의는 표 1과 같다.

표 1. 연구변수의 조작적 정의
Table 1. Operational definitions of research variables

| Construct | Operational definition | Items |
|---|---|---|
| AI trust propensity | Degree to which a developer trusts AI coding tool outputs, recommendations, explanations, and problem-solving ability | TR1 ~ TR4 |
| Verification orientation | Degree to which a developer checks the accuracy, appropriateness, and quality of AI-generated outputs before use | VO1 ~ VO4 |
| AI use frequency | Degree to which a developer frequently uses AI coding tools in development work | UIF1 ~ UIF3 |
| AI dependence | Degree to which a developer relies on AI suggestions, decisions, and code-generation outputs during development | AR1 ~ AR4 |
| Perceived productivity | Degree to which a developer perceives improvement in development speed and work efficiency through AI coding tools | PP1 ~ PP4 |
| Technical debt risk | Degree to which a developer perceives risks in code quality, maintainability, and long-term management due to AI coding tool use | TD1 ~ TD4 |

### 3.2 자료수집 및 표본

본 연구는 AI 코딩 도구를 실제로 사용한 경험이 있는 개발자 또는 개발 관련 업무 수행자를 대상으로 설문조사를 실시하였으며, 총 200부의 응답을 확보하여 분석에 활용하였다. 표본의 일반적 특성은 표 2와 같다.

표 2. 표본의 일반적 특성
Table 2. Demographic characteristics of respondents

| Category | Group | Freq. | Ratio |
|---|---|---|---|
| Gender | Male | 151 | 75.5 |
| | Female | 49 | 24.5 |
| Age | 20s | 83 | 41.5 |
| | 30s | 84 | 42.0 |
| | 40s | 22 | 11.0 |
| | 50s or older | 11 | 5.5 |
| Career | Less than 1 year | 12 | 6.0 |
| | 1 - 3 years | 75 | 37.5 |
| | 3 - 5 years | 27 | 13.5 |
| | 5 - 10 years | 50 | 25.0 |
| | More than 10 years | 36 | 18.0 |
| Role | Front-end development | 31 | 15.5 |
| | Back-end development | 56 | 28.0 |
| | Full-stack development | 42 | 21.0 |
| | Data/AI development | 27 | 13.5 |
| | QA/testing | 13 | 6.5 |
| | DevOps/infrastructure | 21 | 10.5 |
| | PM/PO/planning | 10 | 5.0 |
| Main AI tool | ChatGPT | 48 | 24.0 |
| | GitHub Copilot | 66 | 33.0 |
| | Claude | 32 | 16.0 |
| | Cursor | 33 | 16.5 |
| | Gemini | 17 | 8.5 |
| | Other | 4 | 2.0 |
| Recent AI use | 1 - 2 times per week | 5 | 2.5 |
| | 3 - 4 times per week | 5 | 2.5 |
| | Almost daily | 150 | 75.0 |
| | Several times a day | 40 | 20.0 |

### 3.3 분석 방법

본 연구는 연구가설 검증을 위해 SmartPLS를 활용한 PLS-SEM 분석을 수행하였다. PLS-SEM은 잠재변수 간의 구조적 관계를 분석하는 데 적합하며, 복수의 측정문항으로 구성된 구성개념의 신뢰도와 타당도를 동시에 검토할 수 있다[10][30]. 본 연구는 먼저 측정모형의 신뢰도와 타당도를 평가한 후, 구조모형을 통해 가설을 검증하였다.

측정모형 평가는 외부적재값, Cronbach's α, CR(Composite Reliability), AVE(Average Variance Extracted), HTMT(Heterotrait-Monotrait ratio)를 중심으로 수행하였다[10][11][28]. 구조모형 평가는 inner VIF(Variance Inflation Factor), 경로계수, t값, p값, $R^2$, $f^2$를 중심으로 수행하였다. 또한 AI 의존 경향의 매개역할을 확인하기 위해 specific indirect effects를 추가적으로 분석하였다. Bootstrapping은 5,000회 반복추출을 기준으로 수행하였으며, 가설 채택 여부는 5% 유의수준을 기준으로 판단하였다.

## Ⅳ. 분석 결과

### 4.1 측정모형평가

측정모형의 신뢰도와 타당도를 확인하기 위해 외부적재값, 내적 일관성 신뢰도, 집중타당도 및 판별타당도를 검토하였다. 표 3에 제시된 바와 같이, 모든 측정문항의 외부적재값은 .846 이상으로 나타나 기준치 .70을 상회하였다. 따라서 각 측정문항은 해당 구성개념을 적절히 설명하는 것으로 판단하였다.

표 3. 측정문항의 외부적재값
Table 3. Outer loadings of measurement items

| Construct | Item | Outer loading |
|---|---|---|
| AI trust propensity | TR1 | .863 |
| | TR2 | .945 |
| | TR3 | .948 |
| | TR4 | .921 |
| Verification orientation | VO1 | .921 |
| | VO2 | .950 |
| | VO3 | .923 |
| | VO4 | .919 |
| AI use frequency | UIF1 | .930 |
| | UIF2 | .957 |
| | UIF3 | .946 |
| AI dependence | AR1 | .894 |
| | AR2 | .896 |
| | AR3 | .887 |
| | AR4 | .881 |
| Perceived productivity | PP1 | .905 |
| | PP2 | .958 |
| | PP3 | .961 |
| | PP4 | .954 |
| Technical debt risk | TD1 | .846 |
| | TD2 | .875 |
| | TD3 | .880 |
| | TD4 | .865 |

다음으로 신뢰도와 집중타당도를 검토한 결과, 표 4에 제시된 것과 같이 Cronbach's α는 .889~.960, CR은 .923~.971, AVE는 .751~.893으로 나타났다. 즉, 모든 값이 일반적 기준인 Cronbach's α .70 이상, CR .70 이상, AVE .50 이상을 충족하였다. 따라서 본 연구의 측정모형은 내적 일관성 신뢰도와 집중타당도를 확보했다고 판단하였다.

표 4. 신뢰도 및 집중타당도 분석결과
Table 4. Reliability and convergent validity

| Construct | Cronbach's α | CR | AVE |
|---|---|---|---|
| AI use frequency | .939 | .961 | .892 |
| AI trust propensity | .941 | .956 | .846 |
| AI dependence | .912 | .938 | .791 |
| Verification orientation | .948 | .962 | .862 |
| Technical debt risk | .889 | .923 | .751 |
| Perceived productivity | .960 | .971 | .893 |

판별타당도 검토를 위해 HTMT 값을 확인하였다[11]. 분석 결과, HTMT 전체 값은 .083~.672 범위로 나타났으며, 가장 높은 값은 AI 의존 경향과 지각된 기술부채 위험 간 관계에서 확인되었다. 그러나 이 값 역시 .85 기준을 하회하므로 판별타당도에는 문제가 없는 것으로 판단하였다.

## 4.2 구조모형평가

구조모형 분석에 앞서 다중공선성을 확인하기 위해 inner VIF를 검토하였다. 분석 결과, inner VIF 값은 1.000~1.075 범위로 나타나 기준치 3.3을 하회하였다. 따라서 구조모형의 다중공선성 문제는 없는 것으로 판단하였다.

다음의 표 5는 구조모형의 설명력과 효과크기 분석 결과를 제시하고 있다. 먼저 구조모형의 설명력을 확인한 결과, AI 의존 경향의 $R^2$는 .095, 지각된 생산성의 $R^2$는 .135, 지각된 기술부채 위험의 $R^2$는 .367로 나타났다. AI 의존 경향의 $R^2$는 높지 않은 수준이지만, 이는 AI 의존 경향이 AI 신뢰 성향, 검증 성향, AI 사용 빈도만으로 충분히 설명되기보다는 개발자의 기술 숙련도, 조직 문화, 과업 특성, AI 도구의 종류, 이전 사용 경험 등 다양한 요인의 영향을 받을 수 있음을 시사한다. 따라서 본 연구는 AI 의존 경향의 모든 형성 요인을 포괄적으로 설명하기보다, 검증 성향과 사용 빈도가 AI 의존 경향에 미치는 기본적 구조를 확인하는 데 목적이 있다.

표 5. 설명력 및 효과크기 분석결과
Table 5. R-square and effect size results

| Endogenous variable | $R^2$ | Adjusted $R^2$ |
|---|---|---|
| AI dependence | .095 | .081 |
| Perceived productivity | .135 | .131 |
| Technical debt risk | .367 | .363 |
| **Path** | **$f^2$** | **Effect size** |
| AI trust propensity → AI dependence | .003 | None |
| Verification orientation → AI dependence | .038 | Small |
| AI use frequency → AI dependence | .066 | Small |
| AI dependence → Perceived productivity | .156 | Medium |
| AI dependence → Technical debt risk | .579 | Large |

## 4.3 가설검증 결과

가설검증을 위해 bootstrapping 분석을 수행하였으며, 전체적인 구조모형의 분석결과가 다음의 표 6에 정리되어 있다. 분석 결과, AI 신뢰 성향은 AI 의존 경향에 유의한 영향을 미치지 않았다(β=.051, t=.637, p=.524). 따라서 H1은 기각되었다. 반면 검증 성향은 AI 의존 경향에 유의한 부(-)의 영향을 미쳤으며(β=-.188, t=2.871, p=.004), AI 사용 빈도는 AI 의존 경향에 유의한 정(+)의 영향을 미쳤다(β=.252, t=3.849, p<.001). 따라서 H2와 H3은 채택되었다. 또한 AI 의존 경향은 지각된 생산성에 유의한 정(+)의 영향을 미쳤으며(β=.367, t=6.453, p<.001), 지각된 기술부채 위험에도 유의한 정(+)의 영향을 미쳤다(β=.605, t=14.378, p<.001). 따라서 H4와 H5도 채택되었다.

표 6. 구조모형 분석결과
Table 6. Structural model results

| Hypoth-esis | Path | β | t | p | Result |
|---|---|---|---|---|---|
| H1 | AI trust propensity → AI dependence | 0.051 | .637 | .524 | Rejected |
| H2 | Verification orientation → AI dependence | -.188 | 2.871 | .004 | Supported |
| H3 | AI use frequency → AI dependence | .252 | 3.849 | .000 | Supported |
| H4 | AI dependence → Perceived productivity | .367 | 6.453 | .000 | Supported |
| H5 | AI dependence → Technical debt risk | .605 | 14.378 | .000 | Supported |

### 4.4 간접효과 분석

표 7. 간접효과 분석결과
Table 7. Specific indirect effects

| Indirect path | β | t | p | Result |
|---|---|---|---|---|
| AI use frequency → AI dependence → Perceived productivity | .092 | 3.013 | .003 | Sig. |
| AI use frequency → AI dependence → Technical debt risk | .152 | 3.446 | .001 | Sig. |
| AI trust propensity → AI dependence → Perceived productivity | .019 | .620 | .535 | Not Sig. |
| AI trust propensity → AI dependence → Technical debt risk | .031 | .633 | .527 | Not Sig. |
| Verification orientation → AI dependence → Perceived productivity | -.069 | 2.415 | .016 | Sig. |
| Verification orientation → AI dependence → Technical debt risk | -.114 | 2.734 | .006 | Sig. |

추가적으로 AI 의존 경향의 매개효과를 확인하기 위해 표 7에 제시된 바와 같이 간접효과 분석을 수행하였다. 분석 결과, AI 사용 빈도는 AI 의존 경향을 통해 지각된 생산성(β=.092, t=3.013, p=.003)과 지각된 기술부채 위험(β=.152, t=3.446, p=.001)에 유의한 정(+)의 간접효과를 보였다. 반면 검증 성향은 AI 의존 경향을 통해 지각된 생산성(β=-.069, t=2.415, p=.016)과 지각된 기술부채 위험(β=-.114, t=2.734, p=.006)에 유의한 부(-)의 간접효과를 보였다. AI 신뢰 성향의 간접효과는 유의하지 않았다.

## V. 결 론

본 연구는 AI 코딩 도구 활용 맥락에서 개발자의 AI 신뢰 성향, 검증 성향, AI 사용 빈도가 AI 의존 경향에 미치는 영향을 분석하고, AI 의존 경향이 지각된 생산성과 지각된 기술부채 위험에 미치는 영향을 실증적으로 검증하였다. 분석 결과, AI 신뢰 성향은 AI 의존 경향에 유의한 영향을 미치지 않았으나, 검증 성향은 AI 의존 경향을 낮추고 AI 사용 빈도는 AI 의존 경향을 높였다. 또한 AI 의존 경향은 지각된 생산성과 지각된 기술부채 위험에 모두 유의한 정(+)의 영향을 미쳤다.

본 연구의 학술적 시사점은 다음과 같다. 첫째, 본 연구는 AI 코딩 도구 활용의 결과를 단순한 생산성 향상으로만 보지 않고 생산성과 기술부채 위험이라는 양면적 결과로 분석하였다. 둘째, AI 의존 경향을 중심 변수로 설정하여 AI 코딩 도구 사용 경험이 개발자의 성과 인식으로 연결되는 과정을 설명하였다. 셋째, AI 신뢰 성향보다 검증 성향과 사용 빈도가 AI 의존 경향을 더 잘 설명할 수 있음을 실증적으로 제시하였다.

실무적으로는 AI 코딩 도구의 사용 확대와 함께 검증 절차가 병행되어야 한다. 본 연구에서 검증 성향은 AI 의존 경향을 낮추는 것으로 나타났으므로, 조직은 AI 생성 코드에 대한 코드 리뷰, 테스트 자동화, 보안 점검, 리팩토링 기준을 마련할 필요가 있다. 또한 AI 사용 빈도는 AI 의존 경향을 높이고, AI 의존 경향은 생산성과 기술부채 위험을 동시에 높였으므로, 반복적 AI 사용에 수반되는 검증 절차를 제도화해야 한다.

본 연구는 횡단적 설문자료를 활용하였다는 한계가 있으며, AI 의존 경향에 대한 설명력도 제한적이다. 후속 연구에서는 개발자 숙련도, 조직 환경,

과업 복잡도와 위험도, AI 사용 깊이, 조직적 가드레일 등을 포함한 확장모형을 검토할 필요가 있다. 또한 실제 코드 품질 지표, 코드 리뷰 이력, 보안 취약성 데이터와 결합한 연구가 이루어진다면 AI 코딩 도구 활용이 실제 기술부채에 미치는 영향을 더 정교하게 분석할 수 있을 것이다.

## References


[1] A. Sergeyuk, Y. Golubev, T. Bryksin, and I. Ahmed, "Using AI-based coding assistants in practice: State of affairs, perceptions, and ways forward", Information and Software Technology, Vol. 178, Art. no. 107610, Feb. 2025. https://doi.org/10.1016/j.infsof.2024.107610.

[2] A. Moradi Dakhel, V. Majdinasab, A. Nikanjam, F. Khomh, M. C. Desmarais, and Z. M. Jiang, "GitHub Copilot AI pair programmer: Asset or liability?", Journal of Systems and Software, Vol. 203, Art. no. 111734, Sep. 2023. https://doi.org/10.1016/j.jss.2023.111734.

[3] A. Ziegler, E. Kalliamvakou, X. A. Li, A. Rice, D. Rifkin, S. Simister, G. Sittampalam, and E. Aftandilian, "Measuring GitHub Copilot's impact on productivity", Communications of the ACM, Vol. 67, No. 3, pp. 54-63, Mar. 2024. https://doi.org/10.1145/3633453.

[4] J. Butler, J. Suh, S. Haniyur, and C. Hadley, "Dear Diary: A randomized controlled trial of Generative AI coding tools in the workplace", arXiv preprint arXiv:2410.18334, Oct. 2024. https://doi.org/10.48550/arXiv.2410.18334.

[5] S. Barke, M. B. James, and N. Polikarpova, "Grounded Copilot: How programmers interact with code-generating models", Proceedings of the ACM on Programming Languages, Cascais, Portugal, Vol. 7, No. OOPSLA1, Art. no. 78, Oct. 2023. https://doi.org/10.1145/3586030.

[6] M. Waseem, A. Ahmad, K.-K. Kemell, J. Rasku, S. Lahti, K. Mäkelä, and P. Abrahamsson, "Vibe Coding in Practice: Flow, Technical Debt, and Guidelines for Sustainable Use", arXiv preprint arXiv:2512.11922, Dec. 2025. https://doi.org/10.48550/arXiv.2512.11922.

[7] V. Pimenova, S. Fakhoury, C. Bird, M.-A. Storey, and M. Endres, "Good Vibrations? A Qualitative Study of Co-Creation, Communication, Flow, and Trust in Vibe Coding", arXiv preprint arXiv:2509.12491, Sep. 2025. https://doi.org/10.48550/arXiv.2509.12491.

[8] R. Sapkota, K. I. Roumeliotis, and M. Karkee, "Vibe coding vs. Agentic coding: Fundamentals and practical implications of agentic AI", arXiv preprint arXiv:2505.19443, May 2025. https://doi.org/10.48550/arXiv.2505.19443.

[9] W. Cunningham, "The WyCash portfolio management system", ACM SIGPLAN OOPS Messenger, Vol. 4, No. 2, pp. 29-30, Dec. 1992. https://doi.org/10.1145/157710.157715.

[10] J. F. Hair, G. T. M. Hult, C. M. Ringle, and M. Sarstedt, "A Primer on Partial Least Squares Structural Equation Modeling (PLS-SEM)", 3rd ed., Sage, Thousand Oaks, CA, 2022.

[11] J. Henseler, C. M. Ringle, and M. Sarstedt, "A new criterion for assessing discriminant validity in variance-based structural equation modeling", Journal of the Academy of Marketing Science, Vol. 43, No. 1, pp. 115-135, Jan. 2015. https://doi.org/10.1007/s11747-014-0403-8.

[12] S. I. Ross, F. Martinez, S. Houde, M. Muller, and J. D. Weisz, "The Programmer's Assistant: Conversational Interaction with a Large Language Model for Software Development", Proceedings of the 28th International Conference on Intelligent User Interfaces, Sydney, New South Wales, Australia, pp. 491-514, Mar. 2023. https://doi.org/10.1145/3581641.3584037.

[13] N. Perry, M. Srivastava, D. Kumar, and D. Boneh, "Do Users Write More Insecure Code with AI Assistants?", Proceedings of the 2023 ACM

SIGSAC Conference on Computer and Communications Security, Copenhagen, Denmark, pp. 2785-2799, Nov. 2023. https://doi.org/10.1145/3576915.3623157.

[14] H. Pearce, B. Ahmad, B. Tan, B. Dolan-Gavitt, and R. Karri, "Asleep at the Keyboard? Assessing the Security of GitHub Copilot's Code Contributions", Communications of the ACM, Vol. 68, No. 2, pp. 96-105, Feb. 2025. https://doi.org/10.1145/3610721.

[15] A. Bacchelli and C. Bird, "Expectations, Outcomes, and Challenges of Modern Code Review", Proceedings of the 35th International Conference on Software Engineering, San Francisco, California, USA, pp. 712-721, May 2013.

[16] P. C. Rigby and C. Bird, "Convergent Contemporary Software Peer Review Practices", Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering, pp. 202-212, Aug. 2013. https://doi.org/10.1145/2491411.2491444.

[17] P. Kruchten, R. L. Nord, and I. Ozkaya, "Technical Debt: From Metaphor to Theory and Practice", IEEE Software, Vol. 29, No. 6, pp. 18-21, Nov.-Dec. 2012. https://doi.org/10.1109/MS.2012.167.

[18] Z. Li, P. Avgeriou, and P. Liang, "A systematic mapping study on technical debt and its management", Journal of Systems and Software, Vol. 101, pp. 193-220, Mar. 2015. https://doi.org/10.1016/j.jss.2014.12.027.

[19] N. Rios, M. G. de M. Neto, and R. O. Spínola, "A tertiary study on technical debt: Types, management strategies, research trends, and base information for practitioners", Information and Software Technology, Vol. 102, pp. 117-145, Oct. 2018. https://doi.org/10.1016/j.infsof.2018.05.010.

[20] P. Vaithilingam, E. L. Glassman, P. Groenwegen, S. Gulwani, A. Z. Henley, R. Malpani, D. Pugh, A. Radhakrishna, G. Soares, J. Wang, and A. Yim, "Towards more effective AI-assisted programming: A systematic design exploration to improve Visual Studio IntelliCode's user experience", Proceedings of the 45th International Conference on Software Engineering: Software Engineering in Practice, Melbourne, Australia, pp. 185-195, May 2023.

[21] N. Forsgren, M.-A. Storey, C. Maddila, T. Zimmermann, B. Houck, and J. Butler, "The SPACE of developer productivity: There's more to it than you think", Queue, Vol. 19, No. 1, pp. 20-48, Mar. 2021. https://doi.org/10.1145/3454122.3454124.

[22] R. Parasuraman and V. Riley, "Humans and automation: Use, misuse, disuse, abuse", Human Factors, Vol. 39, No. 2, pp. 230-253, Jun. 1997. https://doi.org/10.1518/001872097778543886.

[23] K. A. Hoff and M. Bashir, "Trust in automation: Integrating empirical evidence on factors that influence trust", Human Factors, Vol. 57, No. 3, pp. 407-434, May 2015. https://doi.org/10.1177/0018720814547570.

[24] C. Fornell and D. F. Larcker, "Evaluating structural equation models with unobservable variables and measurement error", Journal of Marketing Research, Vol. 18, No. 1, pp. 39-50, Feb. 1981. https://doi.org/10.1177/002224378101800104.

[25] P. M. Podsakoff, S. B. MacKenzie, J.-Y. Lee, and N. P. Podsakoff, "Common method biases in behavioral research: A critical review of the literature and recommended remedies", Journal of Applied Psychology, Vol. 88, No. 5, pp. 879-903, Oct. 2003. https://doi.org/10.1037/0021-9010.88.5.879.

[26] W. W. Chin, "The partial least squares approach to structural equation modeling", Modern Methods for Business Research, G. A. Marcoulides, Ed., Lawrence Erlbaum Associates, Mahwah, NJ, pp. 295-336, Jan. 1998.

## 저자소개


### 정 성 채 (Sungche Jung)

2014년 2월 : 경희사이버대학교 미디어콘텐츠디자인학과(예술학사)
2016년 8월 : 고려대학교 컴퓨터정보통신 대학원 디지털정보.미디어공학과(공학석사)
2024년 3월 ~ 현재 : 국민대학교 비즈니스IT전문대학원 박사과정
2023년 12월 ~ 현재 : 빅픽처인터렉티브 시니어매니저
관심분야 : AI coding tools, human-AI collaboration, software engineering, technical debt, and generative AI

### 안 현 철(Hyunchul Ahn)

1999년 2월 : KAIST 산업경영학과(공학사)
2002년 8월 : KAIST 테크노경영 대학원(경영공학석사)
2006년 8월 : KAIST 테크노경영 대학원(경영공학박사)
2009년 3월 ~ 현재 : 국민대학교 비즈니스IT대학원 교수
관심분야 : 인공지능 응용, 재무정보시스템, CRM, 정보기술수용