

디자인에서 개발까지:

AI를 활용해 역할을 확장하는 디자이너의 심리·기술적 장벽과 개선 방안

김재희¹·임진호^{2*}

¹국민대학교 테크노디자인전문대학원 스마트경험디자인학과 석사과정

²국민대학교 테크노디자인전문대학원 스마트경험디자인학과 교수

From Design to Development: Overcoming Designers' Psychological and Technical Barriers in Role Expansion with AI

Jae-Hee Kim¹ · Jin-Ho Yim^{2*}

¹Master's Course, Kookmin University, Graduate School of Technology Design, Department of Smart Experience Design, Seoul 02707, Korea

²Professor, Kookmin University, Graduate School of Technology Design, Department of Smart Experience Design, Seoul 02707, Korea

[요약]

바이브 코딩이 확산하고 있지만, 디자이너는 심리적 요인과 기술적 요인에 의해 활용이 제약되고 있다. 본 연구는 바이브 코딩 경험이 있는 디자이너(n=231)를 대상으로 설문을 진행하였고, 심리적 장벽과 기술적 장벽을 변수로 구성하여 실제적 활용과 개발 성과에 미치는 영향을 구조방정식으로 분석하였다. 분석 결과, 심리·기술적 장벽 모두 AI 코드 활용도를 낮추었으며, 기술적 장벽의 효과는 더 크게 나타났다. 도구 활용이 늘어날수록 성과 인식 또한 긍정적으로 강화되었다. 따라서 디자이너가 바이브 코딩의 제약을 낮추기 위해서는, AI 툴 설계 과정에서 불안 완화와 통제감 제고를 고려하여 디자이너가 도구를 원활히 활용할 수 있도록 지침이 마련될 필요가 있다.

[Abstract]

Although vibe coding is spreading rapidly, designers' use of AI code-generation tools remains constrained by psychological and technical factors. We surveyed designers with experience in vibe coding (n=231) and modeled psychological and technical barriers as independent variables to examine their effects on tool utilization and perceived outcomes using structural equation modeling. The results show that both psychological and technical barriers significantly reduced AI code use, with technical barriers having the stronger effect. Greater tool use was positively associated with perceptions of improved outcomes and quality. These findings suggest that, to lower barriers to vibe coding, AI tool design should embed mechanisms that alleviate anxiety and enhance users' sense of control, with support from guidelines that help designers apply these tools more effectively.

색인어 : AI 코드 생성, 디자인-개발 융합, 디자이너, 생성형 AI, 바이브 코딩

Keyword : AI Code Generation, Design-Development Integration, Designer, Generative AI, Vibe Coding

<http://dx.doi.org/10.9728/dcs.2025.26.10.2683>



This is an Open Access article distributed under the terms of the Creative Commons Attribution Non-Commercial License (<http://creativecommons.org/licenses/by-nc/3.0/>) which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited.

Received 02 September 2025; **Revised** 24 September 2025

Accepted 10 October 2025

***Corresponding Author, Jin-Ho Yim**

Tel: +82-2-910-6435

E-mail: hci.yim@kookmin.ac.kr

I. 서론

대규모 언어모델(LLM)의 등장은 프로그래밍 방식의 근본적 전환을 야기하고 있다. 특히, 자연어 프롬프트만으로 즉시 실행 가능한 코드를 생성·수정하는 ‘바이브 코딩(vibe coding)’ 패러다임이 주목받고 있다. Copilot 사용자는 ‘탐색(exploration)’과 ‘가속(acceleration)’ 모드를 유연하게 전환하며 아이디어 탐색 및 문제 해결에 도구를 활용하는 것으로 보고되었다[1]. 이는 LLM이 개발자의 인지적 과정과 작업 방식에 실질적인 영향을 미침을 시사한다.

LLM4FaaS 프레임워크는 비전공자도 자연어 프롬프트만으로 서버리스(serverless) 애플리케이션을 배포할 수 있음을 보이며, 노코드·로우코드를 넘어서는 제3세대 개발 방식으로 제안되었다[2]. 국내 연구에서는 LLM 기반 자동 코드 리뷰 학습 환경이 학습자의 자율적 코드 개선을 촉진함을 입증하였다[3]. 이러한 흐름은 바이브 코딩이 생산성 향상을 넘어 교육 현장의 변화를 촉발하고 있음을 보여준다. 또한 Karpathy의 직접 발언에서 ‘vibe coding’ 용어가 제시되었으며, 이후 업계 자료에서 개념이 정리·확장되었다[4],[5]. 본 연구에서는 ‘바이브 코딩’을 자연어 프롬프트를 통해 AI와 상호작용을 하며 코드를 작성·수정·실행하는 행위로 정의한다.

LLM 기반 코드 생성 도구는 진입 장벽을 낮추지만, 심리적·기술적 장벽은 여전히 존재한다. Copilot 사용자에게는 생성 코드의 신뢰성, 이해·수정의 어려움, 통제력 약화 등 심리적 부담이 보고되었고[6], 교육 현장 사례에서는 통제감 부족과 책임 불분명, 할루시네이션 및 출처 불명확성으로 인한 검증·정제의 인지적 부담이 지적되었다[7]. 기술적 측면에서는 생성 코드 디버깅 과정에서 인지 부하, 시선 집중 시간, 탐색 행동이 유의하게 증가하였고, 도구 숙련도가 낮을수록 문제 해결 부담이 가중되는 것으로 나타났다[8]. 이러한 결과는 AI 코드 생성 도구의 도입이 일부 사용자에게 심리적 위축과 기술적 좌절을 유발할 수 있음을 시사한다.

선행 연구는 도구 수준의 효과(생산성) 또는 사용성/행동 보고에 집중되어 있다[1],[9]. 반면 사용자 측 장벽을 다차원으로 측정하고 그것이 사용과 성과 인식에 미치는 영향을 구조방정식으로 검증한 연구는 제한적이다. 특히 디자이너 집단의 바이브 코딩 맥락에서 통합 경로 모형을 제시한 연구는 보고되지 않았다. 본 연구는 이러한 공백을 해결하고자 한다.

연구 대상은 AI 코드 생성 도구를 활용하여 웹·앱 개발을 수행하거나 시도한 디자이너 및 디자인 전공자로 한정하며, 바이브 코딩 상황에서 나타나는 심리적·기술적 장벽에 초점을 둔다. 본 연구는 디자인 프로세스 전반을 다루기보다, 디자이너가 자신이 설계한 결과물을 구현 단계까지 연계하는 과정에서 직면하는 장벽에 주목한다. 이를 통해 디자이너가 바이브 코딩을 더욱 효과적으로 활용할 수 있도록 교육·도구·업무 프로세스 차원의 시사점을 제시하고자 한다.

따라서 본 연구의 목적은 이러한 장벽 요인이 실제 사용과

개발 성과 인식에 어떠한 영향을 미치는지를 실증적으로 분석하는 데 있으며, 이를 구체화하기 위하여 다음과 같은 세 가지 연구 목표를 설정하였다. 첫째, 문헌 검토와 인터뷰를 기반으로 장벽 요인을 도출하고, 이를 토대로 신뢰도·타당도를 갖춘 측정 척도를 개발한다. 둘째, 도출된 요인들이 AI 코드 활용도(USE)와 개발 성과 인식(OUT)에 미치는 영향을 확인하기 위해, 확인적 요인분석(CFA)과 구조방정식 모형(CB-SEM)을 적용하여 직접 및 간접 경로를 검증한다. 셋째, 분석 결과를 바탕으로 디자이너가 설계한 결과물을 구현 단계까지 원활하게 연계할 수 있도록 교육·도구·워크플로 차원의 시사점을 제시한다.

II. 이론적 배경

2-1 AI 코드 생성 도구와 바이브 코딩

LLM은 코드 자동 생성·수정·문서화를 통해 개발 효율을 높인다. 현장·설문 연구에서 생산성과 만족도의 향상 가능성이 반복적으로 보고되었다[9],[10]. 한편, 생성 코드를 검증·수정하는 과정에서 추가 탐색과 인지적 부담이 수반되는 양상이 관찰된다[1],[8]. 더불어 일부 프레임워크는 비전공자도 자연어 프롬프트만으로 서버리스 애플리케이션을 배포할 수 있음을 보여주어, 개발 진입 장벽의 하락 가능성을 시사한다[2].

2-2 디자이너의 개발 참여와 장벽

AI 보조 코딩 도구의 확산은 디자이너의 개발 참여를 UI 시안·프로토타이핑에서 코드 생성·수정 단계로 확장하고 있다. Copilot 등 대화형 도구는 프롬프트 기반 코드 생성·설명·리팩토링을 지원하며, 생산성 향상과 사용 편의에 대한 보고가 축적되어 왔다[9],[10]. 그럼에도 사용자 측면 장벽은 뚜렷하다. 생성 코드의 신뢰성 문제, 이해·수정의 난이도와 통제감 약화가 반복적으로 보고 있다[6]. 또한 검증·디버깅 단계에서는 인지 부하가 증가하고 탐색 행동이 확대되는 양상이 확인된다[8]. 교육·학습 맥락에서도 대규모 수업 배치 결과, AI 보조도구가 학습 속도를 높이는 한편 검증·정제 부담과 과의존 문제가 병존함이 보고된다[11],[12]. 장벽은 심리적 장벽과 기술적 장벽이 함께 작동하는 문제로 이해된다. 심리적 장벽은 불안, 자기효능감, 복잡성 인식, 정체성 저항으로 구성된다. 기술적 장벽은 기초 개념, 도구 숙련, 문제해결 능력으로 구성된다[1],[6],[8]. 또한 업무 정체성 위협은 AI 보조도구 수용 과정에서 저항·회피로 나타날 수 있음이 조직 맥락 연구에서 보고된다[13].

2-3 장벽 요인 측정에 대한 기존 한계

선행연구들을 살펴보면, 생산성이나 만족도 등 도구 차원의 효과 검증이나 사용성 및 경험에 대한 보고 중심의 접근이 주를 이루었다[1],[9],[10]. 반면, 장벽 요인들을 다차원적으로 계량화하고 이를 사용과 성과 인식 간 관계로 연결하여 통합적 모형으로 검증한 연구는 상당히 제한적인 상황이다. 행동 관찰을 통한 연구들에서는 LLM을 활용한 코드 검증 및 디버깅 과정에서 발생하는 인지 부하와 탐색 행동의 증가를 세밀하게 분석하였으나[8], 이러한 다차원적 장벽 요소들을 체계적으로 척도화하고 측정모형과 구조모형을 연계하여 직간접 효과까지 추정하는 분석은 매우 드물다.

해의 연구 동향을 보면, 엔드유저 프로그래밍에서 나타나는 장벽을 설계, 선택, 조정, 사용, 이해, 정보의 6개 범주로 체계화하여 비전문가들의 학습 및 사용 과정에 구조적인 장벽이 존재함을 체계적으로 규명한 바 있다[14]. 아울러 대규모 수업 환경에서의 배치 연구들은 사용상의 이점과 더불어 품질 검증, 책임 소재, 학습 전이 등의 과제를 동시에 제기하고 있다[11],[12]. 최근에는 현업 소프트웨어 엔지니어들을 대상으로 한 무작위 통제 실험 연구가 등장하여 도구 효과에 대한 인과적 검증을 시도하고 있다[15]. 그러나 비개발자나 디자이너 집단을 대상으로 장벽, 사용, 성과 간의 통합적 경로를 분석한 연구는 여전히 부족한 상황이다.

국내 연구 현황을 살펴보면, LLM 기반 학습 환경의 설계 및 경험에 관한 연구 결과들이 꾸준히 축적되고 있다[3],[7]. 하지만 디자이너 집단을 대상으로 바이브 코딩 상황의 심리적·기술적 장벽 구조를 척도화부터 측정모형, 구조모형까지 체계적으로 연계하여 검증한 연구는 부족하다. 따라서 본 연구는 디자이너 표본을 대상으로 장벽 구성개념 도출 → 계량 척도 개발 → 확인적 요인분석(CFA) → 구조방정식 모형(CB-SEM)의 절차를 거쳐, 사용 행동의 매개효과를 실증적으로 검증하고자 한다.

III. 연구 방법

3-1 연구모형 및 가설 설정

1) 연구모형

본 연구에서는 디자이너가 바이브 코딩 과정에서 직면하는 심리적 장벽(Psychological Barriers, PBR)과 기술적 장벽(Technical Barriers, TBR)이 AI 코드 활용도(AI Code Utilization, USE)에 미치는 영향을 설정하였다. 또한, AI 코드 활용도(USE)가 성과 인식(Outcome Perception, OUT)으로 이어지는 경로를 포함하였다. 이를 통해 장벽 요인이 간접적으로 성과 인식에 어떤 영향을 주는지 검증하고자 한다. 그림 1은 본 연구의 연구모형을 도식화한 것으로, 실선은 직접 효과를, 점선은 간접 효과를 의미한다.

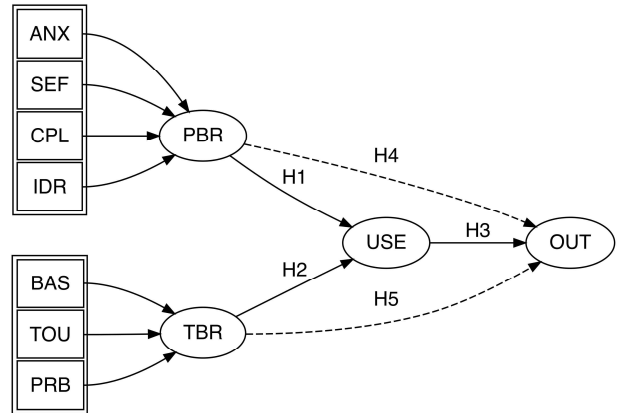


그림 1. 디자이너가 바이브 코딩 과정에서 직면하는 심리적·기술적 장벽(PBR, TBR)이 AI 코드 활용도(USE)와 성과 인식(OUT)에 미치는 영향 구조

Fig. 1. Psychological and technical barriers (PBR, TBR) in vibrate coding and their impact on AI code utilization (USE) and outcome perception (OUT)

2) 가설 설정

장벽과 사용·성과 간의 관계는 다음과 같이 설정한다.

- H1. 심리적 장벽(PBR)은 AI 코드 활용도(USE)에 부(-)의 영향을 미친다.
- H2. 기술적 장벽(TBR)은 AI 코드 활용도(USE)에 부(-)의 영향을 미친다.
- H3. AI 코드 활용도(USE)는 성과 인식(OUT)에 정(+)의 영향을 미친다.
- H4. 심리적 장벽(PBR)은 AI 코드 활용도(USE)를 매개로 성과 인식(OUT)에 부(-)의 간접 영향을 미친다.
- H5. 기술적 장벽(TBR)은 AI 코드 활용도(USE)를 매개로 성과 인식(OUT)에 부(-)의 간접 영향을 미친다.

- PBR(심리적 장벽): ANX(불안), SEF(자기효능감 결여), CPL(복잡성 인식), IDR(정체성 저하)
 - TBR(기술적 장벽): BAS(기초 프로그래밍 이해 부족), TOU(개발 도구 숙련 부족), PRB(문제 해결 능력 부족)
- 이하 본문에서는 편의상 약어(코드)를 사용한다. PBR, TBR, USE, OUT, ANX, SEF, CPL, IDR, BAS, TOU, PRB.

3) 구성 개념 도출 및 이론적 근거

장벽의 1차 차원은 디자이너 대상 반구조화 인터뷰를 개방 코딩으로 분석하여 도출하였다. 반복 출현 빈도와 응집성을 기준으로 PBR: ANX, SEF, CPL, IDR, TBR: BAS, TOU, PRB 등 7개 코드를 확정하였다. 대표 진술로는 "코드가 맞는지 불안하다(ANX)", "혼자 수정할 자신이 없다(SEF)", "환경 설정 과정이 너무 복잡하다(CPL)", "나는 개발자가 아니다(IDR)", "기초 문법이 약하다(BAS)", "IDE나 Git 사용이 낯설다(TOU)", "버그 원인 찾기가 어렵다(PRB)" 등이 있다.

도출된 장벽 요인은 본 연구에서 새롭게 구성된 문항이지

만, 선행연구에서 논의된 개념과도 긴밀히 연결된다. 예를 들어, ANX는 컴퓨터 사용 과정에서 나타나는 불안의 정의와 일치하며[16], SEF는 자기효능감 개념을 기반으로 이해될 수 있다[17]. CPL은 정보시스템 분야에서 다루어진 지각된 용이성과 복잡성 개념과 연결되며[18],[19] IDR은 사용자 저항 및 정체성 관련 연구에서 제기된 논의와 유사하다[13],[20]. 또한 BAS는 프로그래밍 교육에서 꾸준히 지적된 기초 지식 격차와 관련되고[21],[22], TOU는 개발 도구 숙련도의 차이에서 비롯된 어려움과 연결된다[8],[23]. 마지막으로 PRB는 프로그래밍 문제 해결 과정에서의 역량 부족을 지적한 선행 연구들과 부합한다[22],[24]. 따라서 본 연구의 설문 문항은 인터뷰에서 수집된 진술을 토대로 구체화하되, 이러한 선행연구의 개념적 맥락을 함께 반영하여 설계하였다.

3-2 연구 절차

본 연구는 디자이너 20명을 대상으로 한 인터뷰를 기반으로 문항을 도출하고, 파일럿 조사를 통해 척도를 점검한 뒤, 본조사 자료를 수집하여 확인적 요인분석(CFA)과 구조방정식모형(SEM)으로 가설을 검증하였다. 전체 절차는 표 1에 요약하였다.

연구 절차는 인터뷰(n=20) → 문항 개발(CVI) → 파일럿 조사(n=35, EFA·신뢰도) → 본조사(n=231) → CFA(측정모형) → CB-SEM(구조모형) 분석의 순으로 진행되었다.

표 1. 연구 절차 요약

Table 1. Summary of research procedure

Phase	Key Activities	Outcome
Exploratory Interviews	Semi-structured interviews (n=20), open and axial coding (thematic coding)	Extraction of candidate barrier concepts
Item Development	Integration of prior research and qualitative findings, expert evaluation (CVI)	Draft questionnaire (30 items)
Pilot Survey	EFA (principal factor extraction / Promax), reliability analysis (n=35)	Factor structure check, items retained
Main Survey	Quota sampling, online survey, quality filtering (n=231)	Final dataset
Measurement Model Testing	1st and 2nd CFA (MLR), CR / AVE / HTMT assessments	Validation of reliability and model fit
Structural Model Analysis	CB-SEM (MLR, std.lv = TRUE)	Path coefficients and explanatory power (R ²)

3-3 구성개념 및 측정

장벽의 모형화는 2-4절에서 제시한 2차(고차) 구조를 따른다. 각 잠재변수의 하위 요인 및 측정 문항 개수는 표 2에 정리하였다.

표 2. 변수구조

Table 2. Structure of variables

Higher-Order Factor	Lower-Order Factors (Constructs)
Psychological Barriers (PBR)	ANX (Anxiety), SEF (Lack of Self-Efficacy), CPL (Perceived Complexity), IDR (Role Identity Resistance)
Technical Barriers (TBR)	BAS (Lack of Basic Programming Understanding), TOU (Lack of Tool Proficiency), PRB (Lack of Problem-Solving Ability)
AI Code Utilization (USE)	Level of tool usage (5 items)
Outcome Perception (OUT)	Perceived outcomes (e.g., time saving, quality improvement, 4 items)

모든 문항은 7점 리커트 척도(1=전혀 그렇지 않다, 7=매우 그렇다)로 측정하였다. 역문항은 SEF, BAS, TOU, PRB에 포함된 항목에 적용하여 ‘값이 클수록 장벽이 큼’이 일관되게 반영되도록 하였다. 연구에 사용된 30개 본 문항은 표 3에 제시하였다.

3-4 표본 설계 및 자료 수집

표본은 AI 코드 생성 도구(ChatGPT, Claude, Cursor, GitHub Copilot 등)를 활용해 웹/앱 개발을 수행하거나 시도한 경험이 있는 디자이너 및 디자인 전공자로 한정하였다. 패널 기반 목적표집을 통해 온라인 설문을 시행하였으며, 수집된 응답 중 불성실 응답(동일 선택 반복, 과도하게 빠른 응답, 일정 패턴 반복)은 사전 기준에 따라 제거하였다.

최종 유효 표본은 231명이었으며, 평균 연령은 30.5세(표준편차 5.4)였다. 성별은 남성 117명(50.6%), 여성 114명(49.4%)으로 균형 있게 분포하였다. 직무는 UX/UI 디자이너(42.9%), 서비스·프로덕트 디자이너(33.3%), 그래픽/브랜드 디자이너(23.4%), 디자인 전공 학생(0.4%) 순으로 나타났다. 전공은 전원 디자인 계열이었다. AI 코드 생성 도구 사용 경험 기간은 1~3개월(29.9%)이 가장 많았다. 프로그래밍 경험은 복수응답 결과 HTML/CSS(191명), JS/React(90명), Python(90명)이 보고되었으며, 40명은 경험이 없다고 응답하였다. 주 사용 도구는 전원이 ChatGPT를 활용하고 있었고, GitHub Copilot(118명), Cursor(88명), Claude(13명)가 추가로 보고되었다. (프로그래밍 경험과 주 사용 AI 코드 도구는 복수응답 문항으로, 빈도만 제시하고 비율은 표에서 생략하였다) 표본의 인구통계학적 특성은 표 4에 제시하였다.

표 3. 측정 변수 및 설문 문항(최종 30문항)

Table 3. Measurement variables and survey items (final 30 items)

Code	Sample Item	Reverse
ANX1	"I feel anxious that the AI-generated code may contain errors."	
ANX2	"I feel tense when checking the results of AI-generated code."	
ANX3	"I am worried that the AI-generated code may not work as expected."	
SEF1	"I have no confidence in modifying AI-generated code."	✓
SER2	"I struggle to come up with ways to utilize AI-generated code effectively."	✓
SEF3	"I find it difficult to understand AI-generated code without assistance."	✓
CPL1	"Using AI-generated code makes the development process overly complicated."	
CPL2	"There is too much to learn in order to effectively use AI-generated code."	
CPL3	"The process of creating with AI-generated code feels cumbersome."	
IDR1	"It feels inappropriate for designers to handle AI-generated code."	
IDR2	"Writing and handling code should be the developer's domain."	
IDR3	"Using AI-generated code does not align with my role as a designer."	
BAS1	"I cannot immediately recall basic programming syntax when reading AI-generated code."	✓
BAS2	"I find it difficult to distinguish between variables and functions in AI-generated code."	✓
BAS3	"It is hard to follow conditional flows when debugging AI-generated code."	✓
TOU1	"It is difficult to explain the auto-completion features of AI-generated code."	✓
TOU2	"I do not know how to integrate AI-generated code with version control tools."	✓
TOU3	"I often make command errors when running or building AI-generated code in the terminal."	✓
PRB1	"I am anxious that I cannot solve AI code errors using only online resources or documentation."	✓
PRB2	"If unexpected bugs occur while modifying AI-generated code, I cannot resolve them."	✓
PRB3	"Testing and running AI-generated code feels overly complex and burdensome."	✓
USE1	"I regularly use AI code generation features in my projects."	
USE2	"I apply AI-suggested code immediately by executing or editing it."	
USE3	"I feel that using AI-generated code speeds up my work."	
USE4	"I have a strong intention to continue using AI-generated code in the future."	
USE5	"I plan to prioritize AI code generation features in future projects."	
OUT1	"I am satisfied with the quality of outcomes generated using AI code."	
OUT2	"AI-generated code has reduced my working time."	
OUT3	"I feel that I have learned and grown through using AI-generated code."	
OUT4	"When I shared the results of AI-generated code, others responded positively."	

표 4. 응답자 특성(n=231)

Table 4. Sample characteristics (n=231)

Category	Items	n	%
Gender	Male	117	50.6
	Female	114	49.4
Age	Mean (SD)	30.5 (5.4)	-
Job	UX/UI Designer	99	42.9
	Service/Product Designer	77	33.3
	Graphic/Brand Designer	54	23.4
	Design-major Student	1	0.4
Major	Design-related	231	100.0
AI Tool Experience	Less than 1 month	24	10.4
	1-3 months	69	29.9
	3-6 months	59	25.5
	6-12 months	57	24.7
	More than 12 months	22	9.5
Programming Experience	None	40	-
	HTML/CSS	191	-
	JS/React	90	-
	Python	90	-
Main AI Tool Used	ChatGPT	231	-
	GitHub Copilot	118	-
	Cursor	88	-
	Claude	13	-

3-5 문항 개발 및 파일럿 조사

인터뷰 전사본은 개방 코딩으로 의미 단위를 라벨링 한 뒤 코드 축약(범주화) 하여 유사·중복 코드를 통합하였다. 이 과정을 통해 ANX, SEF, CPL, IDR, BAS, TOU, PRB 등 장벽 후보를 도출하고, 선행연구 검토와 전문가 평가(CVI)를 거쳐 설문 문항을 확정하였다. 파일럿 조사(n=35)에서는 EFA(주축요인추출, Promax 회전) 과 신뢰도 점검을 실시하였으며, 요인 구조와 내적 일관성이 권고 기준을 충족하여 본조사 문항을 그대로 유지하였다.

3-6 측정모형 검증

측정모형은 2차(고차) 반영형 구조를 전제로 수행하였다. 우선 1차 구인(ANX, SEF, CPL, IDR, BAS, TOU, PRB, USE, OUT)의 수렴·판별타당도를 점검하고, 이어 상위 장벽 구인(PBR, TBR)의 타당도를 2차 CFA로 추가 검증하였다. 평가는 다음 기준에 따른다: 전역 적합도(CFI/TLI, RMSEA/SRMR), 수렴타당도(표준화 적재량 권고 ≥ 0.70 , 허용 ≥ 0.60 ; CR ≥ 0.70 ; AVE ≥ 0.50), 판별타당도(대각선 $\sqrt{AVE} > \text{교차상관}$, HTMT < 0.85). 상위 적재(PBR/TBR \rightarrow 하위 요인)는 양(+)이어야 한다. 세부 수치는 4장에서 보고한다.

3-7 구조모형 분석

구조모형은 PBR → USE, TBR → USE, USE → OUT의 경로로 설정하였다(상위 장벽 간 상관 허용). 추정은 CB-SEM(lavaan, MLR, std.lv = TRUE)으로 수행하며, 전역 적합도, 경로계수(β , z, p) 및 설명력(R^2)을 보고한다. PBR과 TBR의 상대적 영향은 Wald 제약 검정(PBR→USE = TBR→USE)으로 확인한다. 간접효과(PBR/TBR → USE → OUT)는 부트스트랩(5,000회, 양측) 95% 신뢰구간과 함께 4장에서 제시하였다.

IV. 연구 결과

4-1 기술통계

30개 문항에 대한 평균, 표준편차, 왜도 및 첨도를 산출하였다. 모든 문항은 1-7 척도 범위 내에서 적절히 분포하였으며, 왜도와 첨도 값이 절댓값 2.0 이내로 나타나, 정규성에 큰 문제가 없음을 확인하였다. 또한 구성개념 수준의 상관 분석 결과, PBR과 USE 간에는 정(+)의 관계가 나타났으나 이는 역채점에 따른 부호 반전으로 개념상 부(-)의 관계와 일치하였다. USE와 OUT 간에는 정(+)의 관계가 관찰되어 가설의 방향성과 일치하였다. 세부 수치는 문항별 기술통계 표 5와 구성개념 상관행렬 표 6에 제시하였다.

4-2 측정모형 결과 (2차 반영형)

확인적 요인분석(CFA)에서 최종 2차(고차) 반영형 측정모형의 전반적 적합도는 수용 기준을 충족하였다. 1차 구인의 표준화 적재량은 권고치(≥ 0.70 , 허용 ≥ 0.60)를 전반적으로 상회하였고, 신뢰도·수렴타당도 지표(Cronbach's α , CR, AVE) 역시 권고치를 만족하였다. 판별타당도는 대각선 \sqrt{AVE} 가 모든 교차상관보다 크고, HTMT가 0.85 미만으로 확인되었다. 세부지표는 표 7에 제시하였다.

4-3 구조모형 결과

구조모형 결과는 two-stage 강건추론에 근거하여 보고한다. 구체적으로, 1단계에서 측정모형을 바탕으로 잠재점수를 산출하고, 2단계에서 HC3 표준오차를 적용한 회귀를 통해 표준화계수(β)를 추정하였다. 부트스트랩 5,000회(양측)로 95% 신뢰구간을 산출하였으며, 유의성 판단은 $p < .05$ 기준을 따른다. 해석하면, H1 검증 결과 심리적 장벽(PBR)은 USE에 부(-)의 영향을 미쳐 가설이 지지되었다($\beta = -0.305$, $p < .001$). H2 검증 결과 기술적 장벽(TBR) 역시 USE에 유의한 부(-)의 영향을 미쳤으며($\beta = -0.466$, $p < .001$), 그 효과 크기는 PBR보다 상대적으로 크게 나타났다. H3 검증 결과 USE는 OUT에 정(+)의 영향을 주는 것으로 확인되었으며($\beta = 0.829$, $p < .001$), 경로계수 크기 역시

가장 높아 도구 활용이 성과 인식에 미치는 핵심 매개임이 드러났다. 또한 설명력 측면에서 USE는 중간 이상, OUT은 높은 수준을 보여, 모형 전반이 설득력 있는 설명력을 확보한 것으로 판단된다. 개별 수치(β , SE, t, p)는 표 8에, 주요 경로 계수와 설명력(R^2)은 그림 2에 제시하였다.

4-4 간접효과 및 경로차 검증

간접효과는 부트스트랩 5,000회(양측)로 산출된 95% 신뢰구간을 기준으로 검증하였다. 분석 결과, PBR은 USE를 매개로 OUT에 유의한 부(-)의 간접효과를 보여 H4가 지지되었고, TBR 또한 유의한 부(-)의 간접효과를 보여 H5가 지지되었다. 특히 TBR → USE → OUT 경로의 부정적 효과 크기가 더 커, 기술적 장벽이 성과 인식 저해에 있어 더욱 직접적인 제약 요인임이 확인되었다. 이는 디자이너들이 바이브 코딩을 수행할 때 기초 지식 및 도구 숙련도의 부족이 심리적 요인보다 더 큰 영향을 미친다는 점을 시사한다. 간접효과와 경로차의 구체적 수치 및 신뢰구간은 표 9에 제시하였다.

표 5. 문항별 기술통계

Table 5. Descriptive statistics by item

Variable	M	SD	Sk	Ku
ANX1	3.71	1.40	0.01	-0.45
ANX2	3.80	1.41	0.02	-0.40
ANX3	3.77	1.41	-0.03	-0.21
SEF1	4.19	1.58	-0.13	-0.84
SEF2	4.16	1.60	-0.04	-0.87
SEF3	4.24	1.58	-0.10	-0.74
CPL1	3.94	1.35	-0.02	-0.16
CPL2	3.89	1.33	0.04	-0.47
CPL3	3.95	1.35	-0.05	-0.31
IDR1	3.89	1.37	-0.04	-0.34
IDR2	3.87	1.40	0.04	-0.32
IDR3	3.90	1.37	0.09	-0.58
BAS1	4.23	1.50	-0.18	-0.50
BAS2	4.27	1.46	-0.20	-0.54
BAS3	4.32	1.50	-0.16	-0.52
TOU1	4.05	1.63	-0.06	-0.81
TOU2	4.09	1.64	-0.09	-0.87
TOU3	4.01	1.67	-0.10	-0.88
PRB1	4.09	1.27	-0.07	-0.26
PRB2	4.14	1.25	0.01	-0.26
PRB3	4.09	1.20	0.06	-0.28
USE1	3.62	1.32	-0.02	-0.50
USE2	3.55	1.32	-0.08	-0.44
USE3	3.56	1.32	0.07	-0.44
USE4	3.58	1.34	0.03	-0.44
USE5	3.59	1.36	0.06	-0.61
OUT1	3.76	1.37	0.08	-0.49
OUT2	3.78	1.33	0.03	-0.37
OUT3	3.77	1.38	-0.08	-0.45
OUT4	3.80	1.33	-0.13	-0.58

4-5 소결

가설 검증 결과를 종합하면, H1~H5 모두 유의하게 지지되었다. 즉, PBR과 TBR은 모두 USE에 부정적 영향을 미쳤으며, 이들 요인은 USE를 매개로 OUT에 간접효과를 주는 것으로 나타났다. 특히 TBR은 PBR보다 상대적으로 더 큰 영향을 보여, 디자이너들이 직면하는 실제 제약이 기술적 요인에 더 집중되어 있음을 시사한다. 반면 USE는 OUT에 강한 정(+)의 직접효과를 나타내어, 도구 활용이 성과 인식의 핵심 매개임이 실증적으로 검증되었다.

따라서 본 연구의 구조모형은 그림 1에서 제시한 관계가 경험적으로 타당함을 확인하였으며, 구체적 경로계수와 설명력은 그림 2에 제시하였다.

표 6. 구성개념 상관행렬

Table 6. Correlation matrix of constructs

	ANX	SEF	CPL	IDR	BAS	TOU	PRB	USE	OUT
ANX	1.0	-0.04	0.24	0.09	0.01	-0.06	-0.2	-0.41	-0.48
SEF	-0.04	1.0	-0.19	-0.13	0.06	0.04	0.18	0.46	0.47
CPL	0.24	-0.19	1.0	0.14	-0.05	-0.05	0.0	-0.42	-0.47
IDR	0.09	-0.13	0.14	1.0	0.05	-0.1	0.01	-0.37	-0.41
BAS	0.01	0.06	-0.05	0.05	1.0	-0.01	0.07	0.23	0.22
TOU	-0.06	0.04	-0.05	-0.1	-0.01	1.0	-0.05	0.28	0.37
PRB	-0.2	0.18	0.0	0.01	0.07	-0.05	1.0	0.33	0.34
USE	-0.41	0.46	-0.42	-0.37	0.23	0.28	0.33	1.0	0.83
OUT	-0.48	0.47	-0.47	-0.41	0.22	0.37	0.34	0.83	1.0

표 7. 신뢰도·수렴·판별타당도 요약

Table 7. Summary of reliability and validity

Construct	No. of Items	Cronbach's α	CR	AVE	Factor Loading (Min)	Factor Loading (Max)
ANX	3	0.96	0.98	0.93	0.96	0.97
SEF	3	0.97	0.98	0.95	0.97	0.98
CPL	3	0.96	0.98	0.93	0.95	0.97
IDR	3	0.96	0.97	0.93	0.94	0.97
BAS	3	0.96	0.98	0.93	0.96	0.97
TOU	3	0.97	0.98	0.95	0.97	0.98
PRB	3	0.95	0.97	0.92	0.95	0.96
USE	5	0.98	0.98	0.92	0.95	0.97
OUT	4	0.97	0.98	0.92	0.96	0.97

표 8. 구조모형 경로계수(두-스테이지 HC3 기준)

Table 8. Structural path coefficients (two-stage with HC3)

Path	β	SE(HC3)	t	p
PBR → USE	-0.305	0.061	-5.00	<.001
TBR → USE	-0.466	0.057	-8.18	<.001
USE → OUT	0.829	0.034	24.38	<.001

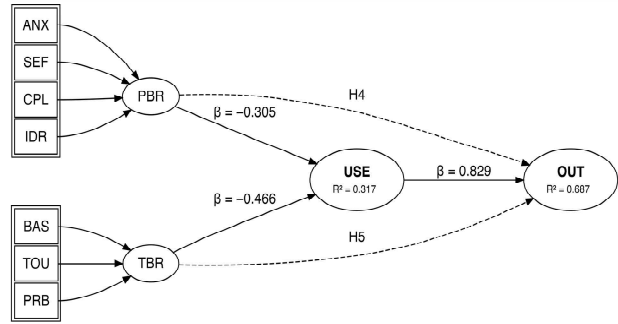


그림 2. 구조모형 분석 결과(β 계수 및 R^2 포함)

Fig. 2. Results of the structural model analysis (including β coefficients and R^2 values)

표 9. 간접효과 및 경로차검정 결과

Table 9. Indirect effects and path-difference test

Effect	Estimate	SE (HC3)	95% CI	Significance
PBR → USE → OUT	-0.253	0.052	[-0.353, -0.147]	Significant
TBR → USE → OUT	-0.387	0.051	[-0.468, -0.308]	Significant
(PBR → USE) - (TBR → USE)	-0.772	0.064	[-0.918, -0.621]	Significant

V. 결론

본 연구는 디자이너가 바이브 코딩 과정에서 경험하는 심리적·기술적 장벽이 AI 코드 활용도(USE)와 성과 인식(OUT)에 어떠한 영향을 미치는지를 실증적으로 규명하였다. 2차 요인 구조를 기반으로 한 측정모형 검증 결과, 심리적 장벽(PBR)과 기술적 장벽(TBR)은 모두 신뢰도와 타당도를 충족하였다. 구조모형 분석에서는 PBR과 TBR이 모두 USE에 부정적 영향을 주었으며, 특히 TBR의 영향력이 PBR보다 상대적으로 크게 나타났다. 한편, AI 코드 생성 도구가 비전문가도 손쉽게 개발할 수 있는 환경을 제공한다는 일반적 인식과 달리, 본 연구에서는 TBR이 PBR보다 더 큰 제약 요인으로 나타났다. 이는 도구의 자동화 수준이 높아졌음에도 불구하고, 실제 사용 과정에서는 환경 설정, 오류 수정, 코드 이해 등 기술적 요구가 여전히 존재함을 시사한다. 즉, ‘누구나 코딩할 수 있다’는 담론과 달리, 디자이너가 AI를 활용해 생성한 코드를 실무적으로 활용하기 위해서는 기초적인 개발 맥락에 대한 이해와 절차적 지원이 필요함을 보여준다. 또한 USE에서 OUT으로 이어지는 경로는 가장 큰 정(+)의 효과를 보여, 도구 활용이 성과 인식의 핵심 매개임이 확인되었다. 이러한 결과는 장벽 요인이 USE를 매개로 OUT에 간접효과를 미친다는 점을 뒷받침하며, 추정 방법 변경에도 일관되게 유지되었다.

정량 분석에서 확인된 이러한 결과는 인터뷰 자료와도 맥락적으로 일치하였다. 인터뷰 응답자들은 공통적으로 ① 오류

수정 능력 부족, ② 전체 프로세스 이해의 어려움, ③ 개발 환경 설정의 어려움을 반복적으로 언급하였는데, 이는 TBR의 하위 요인인 BAS·TOU·PRB와 직접적으로 연결된다. 또한 일부 응답자들은 “코드를 이해하지 못해 생성된 결과의 타당성을 판단하기 어렵다”, “수정 과정에서 어디가 오류인지 파악하기 힘들다”는 불안을 표현하였는데, 이는 PBR의 ANX·SEF 차원을 반영한다. 반면 “반복 요소를 신속하게 생성해 효율이 높아졌다”, “기초 구조를 빠르게 구현하며 성취감을 느꼈다”와 같은 진술은 USE가 OUT으로 이어지는 긍정적 효과를 잘 보여준다. 요컨대, 설문과 인터뷰가 상호 보완적으로 작동하며 본 연구의 주요 결과를 일관되게 지지하였다.

이러한 근거를 바탕으로 실무적 시사점을 제안하면 다음과 같다. 첫째, 교육적 측면에서 프롬프트 작성 훈련보다 코드 해석·부분 수정·오류 대응 역량을 강화하는 단계적 프로그램이 필요하다. 특히 “변경 전후 비교”나 “단위 테스트 통과 과제” 같은 훈련은 BAS·TOU·PRB를 낮추고 SEF를 끌어올리는 효과가 있다. 둘째, 도구 설계 측면에서는 결과의 추론 근거 제시, 오류 위치 하이라이트, 수정 제안, 변경 이력 표시, 자동 테스트 및 롤백 제공과 같은 기능이 중요하다. 이는 CPL·ANX를 완화하고 안정적인 USE를 지원할 수 있다. 셋째, 업무 프로세스 측면에서는 생성·검증·수정 단계를 분리하여 초기에는 저위험·가시성이 높은 과업(예: UI 컴포넌트, 반복 패턴)부터 적용하고 점차 범위를 확장하는 전략이 효과적이다.

한편, 본 연구에는 몇 가지 한계가 존재한다. 첫째, 횡단 설계와 자기보고식 측정에 기반하였기 때문에, 향후 연구에서는 사용 로그(프롬프트 수, 실행/오류 횟수 등)와 코드 품질 지표를 결합한 종단적 검증이 필요하다. 둘째, 표본이 국내 20-40대 디자이너 경험자에게 편중되었으므로 연령대와 경험 수준(미경험자 포함)을 다양화할 필요가 있다. 셋째, 도구 유형별 비교(ChatGPT, Copilot, Cursor 등)를 통해 일반화 가능성을 점검해야 한다.

종합하면, 본 연구는 디자이너의 바이브 코딩 과정에서 심리적·기술적 장벽 → USE → OUT으로 이어지는 경로를 실증적으로 규명하였다. 특히 TBR이 PBR보다 더 큰 제약 요인이 드러났으며, USE는 OUT으로 이어지는 핵심 매개로 확인되었다. 이러한 결과는 교육·도구·업무 프로세스를 아우르는 개선 방안 마련에 실질적 근거를 제공하며, 후속 연구에서는 구체적 실행 전략과 적용 절차를 정교화하는 작업이 요구된다.

참고문헌

[1] S. Barke, M. B. James, and N. Polikarpova, “Grounded Copilot: How Programmers Interact with Code-Generating Models,” *Proceedings of the ACM on Programming Languages*, Vol. 7, No. OOPSLA1, pp. 85-111, April 2023.

<https://doi.org/10.1145/3586030>

[2] M. Wang, T. Pfandzelter, T. Schirmer, and D. Bermbach, “LLM4FaaS: No-Code Application Development using LLMs and FaaS,” arXiv:2502.14450, February 2025 <https://doi.org/10.48550/arXiv.2502.14450>

[3] S. Choi, D.-G. Lee, J. Kim, Y. Jang, and H. Kim, “Designing LLM-Based Code Reviewing Learning Environment for Programming Education,” *The Journal of Korean Association of Computer Education*, Vol. 26, No. 5, pp. 1-11, 2023.

[4] A. Karpathy. “There’s a New Kind of Coding I Call ‘Vibe Coding,’ Where You Fully Give in to the Vibes...,” [Internet]. Available: <https://x.com/karpathy/status/1886192184808149383>.

[5] Google Cloud. What Is Vibe Coding? [Internet]. Available: <https://cloud.google.com/discover/what-is-vibe-coding>.

[6] P. Vaithilingam, T. Zhang, and E. L. Glassman, “Expectation vs. Experience: Evaluating the Usability of Code Generation Tools Powered by Large Language Models,” in *Proceedings of the CHI Conference on Human Factors in Computing Systems Extended Abstracts*, New Orleans: LA, pp. 1-7, April 2022. <https://doi.org/10.1145/3491101.3519665>

[7] S. A. Kyun and S. J. Kwon, “‘Human-AI’ Collaboration in Learning: The Role of Conversational Generative AI and Student Experiences,” *The Journal of Educational Research*, Vol. 22, No. 2, pp. 97-122, 2024. <https://doi.org/10.31352/JER.22.2.97>

[8] N. Tang, M. Chen, Z. Ning, A. Bansal, Y. Huang, C. McMillan, and T. Jia-Jun Li, “A Study on Developer Behaviors for Validating and Repairing LLM-Generated Code Using Eye Tracking and IDE Actions,” arXiv:2405.16081, May 2024 <https://doi.org/10.48550/arXiv.2405.16081>

[9] S. Peng, E. Kalliamvakou, P. Cihon, and M. Demirel, “The Impact of AI on Developer Productivity: Evidence from GitHub Copilot,” arXiv:2302.06590, February 2023. <https://doi.org/10.48550/arXiv.2302.06590>

[10] J. T. Liang, C. Yang, and B. A. Myers, “A Large-Scale Survey on the Usability of AI Programming Assistants: Successes and Challenges,” in *Proceedings of the 46th IEEE/ACM International Conference on Software Engineering (ICSE 2024)*, Lisbon: Portugal, pp. 1-13, 2024. <https://doi.org/10.1145/3597503.3608128>

[11] M. Kazemitabaar, R. Ye, X. Wang, A. Z. Henley, P. Denny, M. Craig, and T. Grossman, “CodeAid: Evaluating a Classroom Deployment of an LLM-Based Programming Assistant that Balances Student and Educator Needs,” in

Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems (CHI '24), Honolulu: HI, pp. 1-20, May 2024. <https://doi.org/10.1145/3613904.3642773>

[12] H. Xu, W. Gan, Z. Qi, J. Wu, and P. S. Yu, "Large Language Models for Education: A Survey," arXiv:2405.13001, May 2024. <https://doi.org/10.48550/arXiv.2405.13001>

[13] M. Mirbabaie, S. Stieglitz, F. Brünker, and N. R. J. Möllmann Frick, "The Rise of Artificial Intelligence—Understanding the AI Identity Threat at the Workplace," *Electronic Markets*, Vol. 32, pp. 73-99, 2022. <https://doi.org/10.1007/s12525-021-00496-x>

[14] A. J. Ko, B. A. Myers, and H. H. Aung, "Six Learning Barriers in End-User Programming Systems," in *Proceedings of the IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC 2004)*, Rome: Italy, pp. 199-206, September 2004. <https://doi.org/10.1109/VLHCC.2004.47>

[15] J. Butler, J. Suh, S. Haniyur, and C. Hadley, "Dear Diary: A Randomized Controlled Trial of Generative AI Coding Tools in the Workplace," arXiv:2410.18334, October 2024 <https://doi.org/10.48550/arXiv.2410.18334>

[16] R. K. Heinssen Jr., C. R. Glass, and L. A. Knight, "Assessing Computer Anxiety: Development and Validation of the Computer Anxiety Rating Scale (CARS)," *Computers in Human Behavior*, Vol. 3, No. 1, pp. 49-59, 1987.

[17] D. R. Compeau and C. A. Higgins, "Computer Self-Efficacy: Development of a Measure and Initial Test," *MIS Quarterly*, Vol. 19, No. 2, pp. 189-211, 1995.

[18] F. D. Davis, "Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology," *MIS Quarterly*, Vol. 13, No. 3, pp. 319-340, 1989.

[19] E. M. Rogers, *Diffusion of Innovations*, 5th ed. New York, NY: Free Press, 2003.

[20] H.-W. Kim and A. Kankanhalli, "Investigating User Resistance to Information Systems Implementation: A Status Quo Bias Perspective," *MIS Quarterly*, Vol. 33, No. 3, pp. 567-582, 2009.

[21] A. Robins, J. Rountree, and N. Rountree, "Learning and Teaching Programming: A Review and Discussion," *Computer Science Education*, Vol. 13, No. 2, pp. 137-172, 2003.

[22] A. Gomes and A. J. Mendes, "Learning to Program—Difficulties and Solutions," in *Proceedings of the International Conference on Engineering Education (ICEE 2007)*, Coimbra: Portugal, 2007.

[23] G. C. Murphy, M. Kersten, and L. Findlater, "How Are Java Software Developers Using the Eclipse IDE?" *IEEE Software*, Vol. 23, No. 4, pp. 76-83, 2006.

[24] M. McCracken, V. Almstrum, D. Diaz, M. Guzdial, D. Hagan, Y. B.-D. Kolikant, ... and T. Wilusz, "A Multi-National, Multi-Institutional Study of Assessment of Programming Skills of First-Year CS Students," in *Proceedings of the Working Group Reports from ITiCSE on Innovation and Technology in Computer Science Education*, Canterbury: UK, pp. 125-180, 2001.



김재희(Jae-Hee Kim)

2024년~현 재: 국민대학교 테크노디자인전문대학원 스마트 경험디자인학과 석사과정

※ 관심분야: AI, Vibe Coding, Interaction Design, UX Design



임진호(Jin-Ho Yim)

2007년: 공학 박사, 고려대학교 산업 시스템정보공학과(제품 및 시스템 개발 전공)

2012년: MBA, 알트대학교(IT Biz & Design Management 전공)

1992년~2009년: 삼성전자 VD(사) 개발팀 책임디자이너
2009년~2013년: 삼성전자 DMC(연) UX센터 수석디자이너
2013년~2016년: SK경영경제연구소 UX Rising Star Forum 위원

2013년~2021년: 삼성전자 의료기기(사) 디자인그룹 수석디자이너

2007년~현 재: 대한인간공학회, 한국HCI학회, UXPA Korea 총신회원

2012년~현 재: UXPA Magazine 편집위원(다국어)

2018년~현 재: (사)대한인간공학회 이사

2022년~현 재: 국민대학교 테크노디자인전문대학원 스마트 경험디자인학과 교수

※ 관심분야: Aging UX Design, Medical & Healthcare UX Design, Human Factors and Ergonomics Design, UX, Engineering, Leisure UX Design