

컴퓨터교육학회 논문지 2026년 제29권 제3호  
<https://doi.org/10.32431/kace.2026.29.3.011>



# AIBlockAgent: 블록 기반 프로그래밍 학습 지원을 위한 AI 에이전트 설계 및 개발 연구

## AIBlockAgent: Design and Development of an AI Agent to Support Block-Based Programming Learning

좌하은<sup>†</sup> · 구덕희<sup>††</sup>

Haeun Choat<sup>†</sup> · Dukhoi Koo<sup>††</sup>

### 요약

본 연구는 블록 프로그래밍 학습에서 초보 학습자가 겪는 인지적 어려움과 교사의 맞춤형 피드백 제공 한계를 해결하기 위해 AI 에이전트(AIBlockAgent)를 설계 및 개발하였다. 교육설계연구(EDR)에 따라 단계적 비계 제공, 실시간 코드 검증, RAG 기반 응답 생성, 정의적 지원의 네 가지 설계 원리를 도출하고 이를 적용한 프로토타입을 구현하였다. 현직 교사 10인을 대상으로 두 차례의 사용성 평가를 실시한 결과, 최종 프로토타입은 1차 평가 대비 오류 대응력과 피드백의 유용성 측면에서 유의미한 점수 향상을 보였다. 본 연구는 AIBlockAgent가 학습자의 자기 주도적 문제 해결을 지원하는 인지적 동반자로서 활용될 가능성을 확인하였으며, 향후 시스템 고도화와 실제 학습자 대상의 실증 연구의 필요성을 제시하였다.

**주제어** 컴퓨터교육, AI 에이전트, 블록 프로그래밍, 교육설계연구, 생성형 인공지능

### ABSTRACT

This study designed and developed an AI agent(AIBlockAgent) to address the cognitive challenges faced by novice learners and the limitations of providing personalized feedback in block-based programming education. Following the Educational Design Research (EDR) framework, four core design principles were derived: phased scaffolding, real-time code verification, RAG-based response generation, and affective support. A prototype was implemented based on these principles, and its usability was evaluated through two iterative rounds involving ten in-service teachers. The results demonstrated that the final prototype achieved significant improvements in error-handling capabilities and feedback usefulness compared to the initial evaluation. This research confirms the potential of AIBlockAgent as a "cognitive partner" that facilitates self-directed problem-solving. Furthermore, the study suggests the necessity for future system enhancements and empirical evaluations involving actual learners to further validate its educational impact.

**Keywords** Computer Education, AI Agent, Block-based Programming, Educational Design Research, Generative AI

†정희원 서울교육대학교 교육전문대학원 컴퓨터 교육전공 박사과정  
 ††중신희원 서울교육대학교 정보교육과 교수(교신 저자)  
 논문투고 2025년 11월 10일  
 심사완료 2026년 01월 13일  
 게재확정 2026년 02월 10일  
 발행일자 2026년 03월 22일

## 1. 서론

디지털 대전환 시대와 인공지능 기술의 발전에 따라 학교 현장에서 프로그래밍 교육의 중요성이 부각되고 있다. 프로그래밍 교육은 학습자의 논리적 사고와 창의적 문제 해결력을 기르는 데 도움이 되며, 학교 현장에서는 학습자의 발달 수준을 고려하여 블록 기반 프로그래밍 언어를 중심으로 수업을 운영하고 있다. 그러나 블록 기반 프로그래밍 언어가 시각적 편의성을 제공함에도 불구하고[1], 초보 학습자들은 프로그래밍으로 문제를 해결하는 과정에서 여러 어려움을 겪는다. 특히 조건문, 반복문, 변수 등 핵심 개념을 이해하고 이를 논리적으로 연결하여 코드로 구현하는 과정에서 높은 인지 부하와 오류를 경험한다.

이러한 문제를 해결하기 위해서는 학습 과정에서 즉각적으로 이루어지는 피드백이 중요하다. 선행 연구에 따르면 문제 해결 과정에서 제공되는 적절한 안내와 피드백은 학습자의 학습 지속성과 몰입을 돕는다[2, 3]. 특히 초보 학습자는 알고리즘을 설계하고, 블록을 탐색하는 전 과정에서 지속적인 지원이 필요하다. 하지만 교사 한 명이 모든 학생에게 실시간으로 맞춤형 피드백을 제공하기는 현실적으로 어렵다. 학급 내에서도 학습자 간의 수준 차이가 크기 때문에, 개별적인 오류를 진단하고 대응하는 데 한계가 있다. 이러한 맞춤형 피드백의 부재는 초보 학습자의 프로그래밍 진입 장벽을 높이는 주요 원인이 된다.

최근 이러한 한계를 극복하기 위한 대안으로 인공지능(AI) 기반 학습 지원 시스템이 주목받고 있다[4, 5]. 대규모 언어 모델(LLM)은 코드 설명, 생성, 번역, 디버깅 등 복잡한 프로그래밍 과제를 수행하며[6, 7] 자연어 프로그래밍의 가능성을 보여주고 있다. 이는 AI가 단순한 보조 도구를 넘어 학습자에게 즉각적인 개별 피드백을 제공할 수 있음을 시사한다. 그러나 기존 시스템은 대부분 텍스트 기반 언어를 지원하거나 단편적인 오류 교정에 머물러 있어, 학습자의 문제 해결 과정을 깊이 있게 파악하는 데 한계가 있다. 반면 AI 에이전트는 학습자의 문제 해결 과정을 인지하고 목표 달성을 위한 최적의 방안을 자율적으로 모색한다. 이는 보조 기능을 넘어 학습자와 능동적으로 상호 작용하면서 실제 교실 수업의 어려움을 보완할 수 있게 한다.

이에 본 연구에서는 생성형 AI의 코드 이해 및 생성 능력과 학습자와의 상호작용 가능성을 기반으로, 초보 학습자의 블록 프로그래밍 학습을 지원하는 AI 에이전트(이하 AIBlockAgent)를 설계하고 개발하였다. 본 연구의 연구 문제는 다음과 같다.

첫째, AIBlockAgent의 핵심 설계 원리는 무엇인가?

둘째, AIBlockAgent는 블록 프로그래밍 학습 지원 도구로서 사용성을 갖추고 있는가?

## 2. 이론적 배경

### 2.1 AI 에이전트

에이전트는 수동적인 도구에 제한되지 않고, 자율적으로 상황을 판단하여 스스로 행동을 선택하는 능력을 갖춘 개체를 의미한다[8, 9]. AI 에이전트는 이러한 에이전트의 개념을 인공지능 분야에 적용한 것으로, 환경과 자율적으로 상호작용을 하며 목표를 달성하는 인공지능 시스템을 말한다[10]. AI 에이전트는 사용자의 입력을 분석하여 행동을 계획하고, 다양한 도구를 활용해 주변 환경과 소통한다[11]. 최근에는 대규모 언어 모델(LLM)을 활용한 AI 에이전트가 주목받고 있다. 이는 언어 모델의 뛰어난 이해력과 인지 능력을 바탕으로 복잡한 문제를 추론하고 해결할 수 있기 때문이다[10, 11].

AI 에이전트는 자율성, 반응성, 주도성, 사회적 능력이라는 네 가지 핵심 속성을 가진다[8]. 첫째, 자율성은 인간의 개입 없이도 스스로 판단하여 행동을 결정하는 능력이다[12]. 이는 명시적인 지시가 없더라도 최적의 해결책을 찾아내는 높은 수준의 사고 과정을 포함한다[13]. 둘째, 반응성은 환경의 변화나 자극을 감지하고 이에 신속하게 대응하는 능력을 뜻한다[14]. 셋째, 주도성은 목표 달성을 위해 스스로 계획을 세우고 능동적으로 행동하는 것을 의미한다. 넷째, 사회적 능력은 언어를 매개로 소통하는 능력으로, 자연어 이해와 생성을 통해 인간과 높은 수준의 상호작용이 가능한 특성을 말한다[15, 16].

본 연구에서는 이러한 AI 에이전트의 특성을 블록 프로그래밍 학습 지원에 적용하였다. 학습자의 현재 프로그래밍 상태를 실시간으로 파악하고, 자율적으로 학습 경로를 판단하여 학습자에게 필요한 피드백을 제시하며, 자연스러운 대화로 소통하는 AIBlockAgent를 개발하고자 하였다.

### 2.2 프로그래밍 학습 지원 AI 관련 연구

최근 생성형 인공지능의 발전은 프로그래밍 학습 환경에 큰 변화를 불러오고 있다. AI를 활용한 학습은 효율성을 높이는 데 크게 기여하지만, 한편으로는 학습자가 AI에 지나치게 의존하여 수동적으로 변할 수 있다는 우려도 제기된다[17]. 따라서 프로그래밍 학습을 지원하는 AI는 학습자의 주도성을 보장하는 세심한 교육적 설계를 기반으로 해야 한다.

Kohen-Vacs et al.(2025)는 AI가 정답을 자동으로 생성하는 기능과 함께 학습자의 사고를 자극하고 문제 해결 과정을 함께 탐색하는 ‘인지적 동반자’로 설계되어야 함을 강조하였다[18]. Denny et al.(2024)는 이러한 인지적 동반자로서 AI가 갖추어야 할 핵심 요소로 세 가지를 제시하였다[19]. 첫째, 비계 제공이다. 정답을 직접 제시하기보다 해결 과정을 안내하는 단계별 지침을 제공해야 한다. 둘째, 적절성이다. 피드백의 내용이 학습자의 현재 수준과 과제 맥락에 맞아야 한다. 셋째, 사용자 경험의 신뢰성이다. 즉각적이고 정확한 응답을 통해 효과적인 학습 경험을 제공해야 한다. 이러한 요소들은 학습자의 인지적 부담을 덜어주는 동시에 자기 주도적인 탐색을 유도하는 역할을 한다.

최근 연구들은 이러한 설계 원리를 기술적으로 구현하고 있다. Ma et al.(2024)는 RAG(Retrieval-Augmented Generation) 기술을 활용해 학습자의 질문에 근거 자료를 바탕으로 피드백을 제공하는 RAGMan 시스템을 개발하였다[20]. Feng & Ghosal(2024)의 CourseAssist는 학습자의 의도를 분류하여 특정 강의 목표에 최적화된 피드백을 생성한다[21].

본 연구에서 개발한 AIBlockAgent는 이러한 선행 연구의 원리를 반영하여 설계하였다. 기존 도구들이 주로 문법 오류를 찾아내거나 결과물을 완성하는 데 초점을 두었다면, AIBlockAgent는 학습자의 현재 상태와 목표 사이의 차이를 분석하여 과정 중심 피드백을 제공한다. 또한 학습자의 질문 횟수나 대화 내용에 따라 피드백의 수준을 조절하여 AI가 ‘인지적 동반자’로서 기능하도록 구현하였다.

### 3. 연구 방법

본 연구는 McKenney & Reeves(2018)가 제시한 교육 설계연구(Educational Design Research, EDR) 절차에 따라 블록 프로그래밍 학습을 지원하는 AI 에이전트를 개발하였다[22]. 연구 과정에서 학습자 지원을 위한 설계 원리를 도출하고 프로토타입을 제작하였으며, 전문가 피드백을 바탕으로 시스템을 반복적으로 개선하였다.

교육 설계연구는 설계기반연구(Design-Based Research, DBR)와 유사하지만, 교수-학습 환경에 초점을 두고 교육적 개입의 효과성과 실천적 유의미성을 강조한다. 이 방법론은 이론적 통찰과 실제적 해결책을 동시에 추구한다는 점에서 의의가 있다[23, 24]. 교육설계연구는 일회성 연구에 그치지 않고 ‘분석 및 탐색’-‘설계 및 구현’-‘평가 및 반성’의 세 단계를 순환적으로 거치며 최적의 해결책을 도출한다. 구체적인 절차는 Fig. 1과 같다.

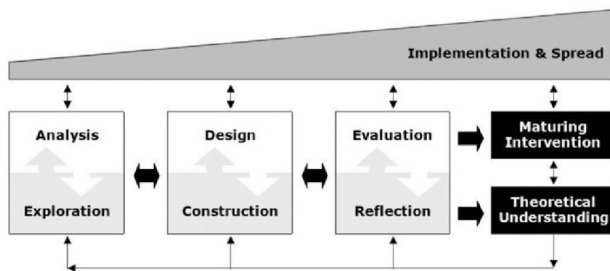


Figure 1. Educational Design Research (McKenney & Reeves, 2018)

본 연구는 교육설계연구 절차에 따라 다음과 같이 진행하였다. 첫째, 분석 및 탐색 단계에서는 블록 프로그래밍 수업을 운영하는 교사들이 경험하는 실질적인 어려움과 요구사항을 파악하여 연구의 목적을 구체화하였다. 둘째, 설계 및 구현 단계에서는 선행 연구 분석을 통해 도출한 내용을 기반으로 AI 에이전트의 핵심 설계 원리를 마련하고 프로토타입

을 개발하였다. 셋째, 평가 및 반성 단계에서는 개발된 시스템의 사용성을 검증하기 위해 두 차례에 걸친 전문가 평가를 실시하였다. 이는 AIBlockAgent가 초보 학습자를 대상으로 하기에 학습자에게 직접 적용하기에 앞서 교육적 설계의 적절성과 사용성을 확인하는 과정이 선행되어야 한다고 판단하였기 때문이다. 이에 블록 프로그래밍 지도 경험이 있는 컴퓨터교육 전공 현직 교사들로 전문가 집단을 구성하였다. 1차 평가에서 수집된 의견을 반영하여 시스템을 고도화하였으며, 2차 평가에서는 동일한 전문가 집단을 대상으로 개선된 시스템의 사용성을 재검증하였다.

## 4. 연구 결과

### 4.1 분석 및 탐색

교육설계연구의 분석 및 탐색 단계는 해결해야 할 교육적 문제를 정의하고 그 원인을 명확히 하는 데 목적이 있다. 이 단계에서는 문헌 조사, 맥락 분석, 요구 분석 등을 통해 교육 현장의 실태와 목표 간의 간극을 확인하고, 이를 해결하기 위한 구체적인 방향을 탐색한다[22].

본 연구에서는 먼저 문헌 조사를 통해 초보 학습자가 블록 프로그래밍 학습에서 겪는 어려움과 이를 지원하기 위한 AI 활용 방안을 분석하였다. 이를 통해 형성적 피드백의 중요성, AI 에이전트의 핵심 속성, 프로그래밍 학습 지원을 위한 방법을 도출하였다. 이어서 현장의 요구를 파악하기 위해 현직 초등교사 8인을 대상으로 블록 프로그래밍 수업 운영 시 겪는 실질적인 어려움에 대한 서술형 설문을 실시하였다.

분석 결과, 교사들은 학습자 간의 수준 차이로 인한 개별 지도의 한계와 학생의 문제 해결 과정에 대한 실시간 피드백의 어려움을 공통적으로 지적하였다. 이러한 요구 분석을 바탕으로, 학습자의 코드를 실시간으로 확인하고 맞춤형 피드백을 제공하는 AI 에이전트의 필요성을 확인하였다. 이는 이후 시스템 설계 단계에서 단계적 비계 제공, 실시간 코드 검증 기반 피드백 생성 등의 설계 원리를 도출하는 근거가 되었다.

### 4.2 시스템 설계 및 구현

#### 4.2.1 시스템 설계

본 연구에서는 블록 프로그래밍 학습을 지원하는 AI 에이전트(AIBlockAgent)를 설계하였다. AIBlockAgent는 정답 코드를 직접 제시하기보다 학습자의 문제 해결 과정을 파악하고 이에 적합한 피드백을 제공하는 데 중점을 두었다. 이를 위해 단계적 비계 제공, 실시간 코드 검증, RAG 기반 응답 생성, 정의적 지원이라는 네 가지 핵심 설계 원리를 도출하였다. 각각의 설계 원리의 세부 내용은 다음과 같다.

첫째, 단계적 비계 제공의 원리이다. 학습자가 동일한 문제에 대해 반복적으로 도움을 요청하는 경우, AI 에이전트는 학습자가 스스로 문제를 해결할 수 있도록 3단계 피드백

구조를 적용한다. 1단계에서는 핵심 알고리즘을 떠올릴 수 있도록 돕는 질문 중심 피드백을 제공한다. 2단계에서는 필요한 블록의 카테고리나 연결 방향을 제시하여 학습 방향을 구체화한다. 마지막 3단계에서는 정답 코드와 구체적인 실행 순서를 안내하여 학습자의 문제 해결을 지원한다. 이러한 점진적 비계 구조는 학습자가 즉각적인 정답에 의존하지 않고 구성주의적 관점에서 지식을 스스로 쌓아가도록 유도한다. 단계별 피드백 예시는 Fig. 2에 제시하였다.

Level	Feedback
1	<p>엔트리봇을 클릭하면 크기가 커지게 하고 싶어요.</p> <p>AI 좋은 아이디어예요! 😊</p> <p>AI 엔트리봇이 클릭했을 때 크기가 커지려면 이런 순서로 생각해 보세요:                  ① 먼저 클릭을 알아채야 해요                  ② 그 다음 크기를 바꿔야 해요</p> <p>'시작' 카테고리에 클릭과 관련된 블록이 있을 거예요. 찾아볼까요?</p>
2	<p>잘 모르겠어요..</p> <p>AI 괜찮아요! 😊 클릭한 것을 알아내려면 '시작' 카테고리의 블록을 사용해야 해요. 그리고 크기를 바꾸려면 '생김새' 카테고리의 블록이 필요해요.</p> <p>AI 1. '시작' 카테고리에서 클릭 관련 블록을 찾아보세요.                  2. '생김새' 카테고리에서 크기를 바꾸는 블록을 찾아보세요.                  3. 그 두 블록을 순서대로 연결해보세요.</p> <p>완성한 후 실행 버튼을 눌러서 확인해보세요!</p>
3	<p>아직도 어려워요. 어떻게 하나요?</p> <p>AI 이렇게 해 보세요! 😊</p> <p>AI 블록 순서:                  1. [오브젝트를 클릭했을 때] ← 시작 블록                  2. [크기를 10%만큼 바꾸기] ← 생김새 블록</p> <p>확인 방법:                  1. 위 순서대로 블록을 쌓아보세요.                  2. 실행 버튼(▶)을 누르세요.                  3. 엔트리봇을 클릭해서 크기가 커지는지 확인하세요.</p> <p>심화 과제: 10 대신 20으로 바꾸면 어떻게 될까요?</p>

Figure 2. Example of the scaffolding feedback (Dialogue illustrating the three levels of AI feedback)

둘째, 실시간 코드 검증의 원리이다. 이는 학습자가 코드를 수정한 뒤 다시 도움을 요청하는 경우 작동한다. AI 에이전트는 학습자의 현재 블록 조립소에 놓인 블록의 종류, 개수, 조립 구조를 분석하여 학습자가 시도한 내용을 파악한다. 올바른 방향으로 나아가고 있다면 긍정적인 격려를, 오류가 발생했다면 이를 교정할 수 있는 힌트를 제공한다. 이를 통해 실시간 평가와 피드백 자동화를 결합하여 학습자의 시행착오를 지원하고자 하였다. Fig. 3은 단계적 비계 제공의 원리와 실시간 코드 검증의 원리가 연계되는 흐름을 나타

낸다.

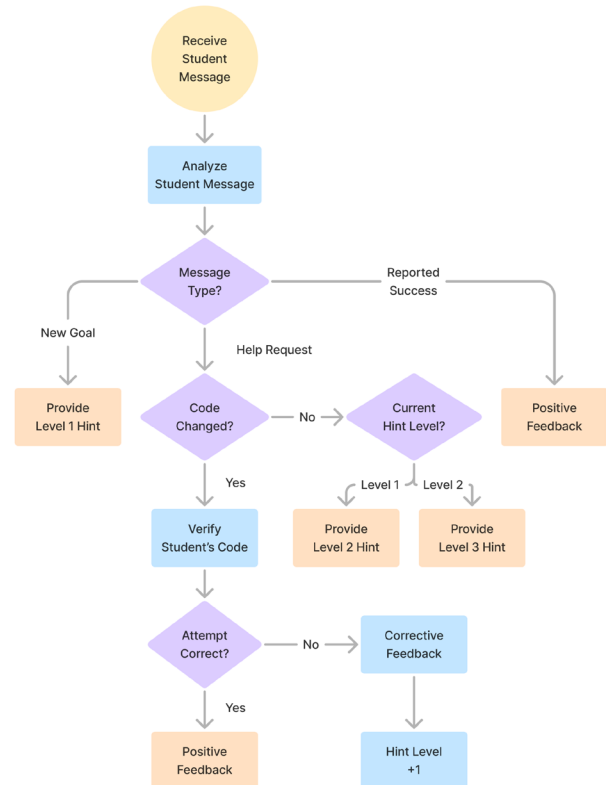


Figure 3. Flow of scaffolding and code verification

셋째, RAG 기반의 응답 생성 원리이다. GPT-4o와 같은 대규모 언어 모델은 뛰어난 추론 능력을 갖추고 있지만, 엔트리 블록의 고유 명칭이나 세부 구조를 정확히 파악하지 못해 존재하지 않는 블록을 제안하는 환각 현상을 일으킨다. 본 연구에서는 이를 해결하기 위해 엔트리 블록의 ID, 이름, 기능 설명, 사용 예시 등을 정리한 마크다운 형식의 BlockDictionary를 구축하고, 이를 AI 에이전트와 연동하였다. BlockDictionary의 세부 항목과 데이터 구성은 Table 1과 같다.

Table 1. Major GAI Services and Types of Responses

Field	Data Content
Block Name	when [KEY] key pressed
Block ID	when_some_key_pressed
Category	Start
Description	Executes the code below when a specific key on the keyboard is pressed.
Parameters	KEY: The keyboard key that triggers the event (e.g., arrow keys, q, w, e).
Default value	when [q] key pressed
Parameter Example	["Up arrow"]
Block Shape	Hat Block
Constraints	Directional arrow keys must be explicitly labeled as 'Up arrow', 'Down arrow', 'Left arrow', and 'Right arrow'

사용자가 질문하면 ContextBuilder가 학습자의 블록 조립소를 실시간으로 탐색하여 현재 사용 중인 블록 정보를 추출한다. 이후 질문과 연관성이 높은 블록 데이터만 선별하여 프롬프트에 동적으로 삽입한다. 이러한 방식은 모든 데이터를 한꺼번에 처리하는 방식에 비해 토큰 소모량을 약 12% 절감하면서 답변의 정확도를 높인다. 동일한 사용자의 질문에 대해 GPT-4o와 AIBlockAgent의 응답을 비교한 결과는 Table 2와 같다.

Table 2. Major GAI Services and Types of Responses

User: I want Entrybot to move in a triangle shape.	
GPT-4o API	When Start button clicked Repeat (3) times: Move forward (100) Turn right (120) degrees
AIBlockAgent	1. When_run_button_click ← Start Block 2. Repeat_basic (3) ← Flow Block └─ Move_direction (100) ← Moving └─ Rotate_relative (120°) ← Moving

GPT-4o 모델은 ‘오른쪽으로 120도 돌기’ 블록을 제안하였으나, 실제 엔트리에는 해당 블록이 존재하지 않는다. 또한, 영어로 응답을 출력하면 ‘이동 방향으로 100만큼 움직이기’ 블록을 ‘Move forward’로 번역한다. 이는 엔트리의 실제 블록명인 ‘Move\_direction’과는 차이가 있어, 언어 모델이 엔트리의 블록을 정확히 인식하고 있지 못함을 알 수 있다. 반면 동일한 GPT-4o 모델을 기반으로 하되 엔트리 블록 정보를 연동한 AIBlockAgent는 ‘방향을 120도만큼 회전하기’ 블록을 제시하여 실제 엔트리 블록명을 사용하는 정확한 피드백을 제공하였다. 다만 GPT-4o 모델은 블록명의 정확도는 낮았으나, 사용자의 프로그래밍 목적을 고려하여 “그림을 그리고 싶다면 ‘펜 내리기’ 블록을 사용할 수 있다.”와 같은 제안을 함께 제공하였다.

셋째, 정의적 측면에서의 지원이다. AI 에이전트는 초보 학습자의 수준에 맞는 어휘를 사용하며 문제 해결 과정에서 긍정적 피드백을 통해 학습 동기를 유지하고 프로그래밍에 대한 부담을 줄이고자 하였다.

본 연구에서 제시한 AIBlockAgent는 이러한 네 가지 설계 원리를 기반으로 초보 학습자의 블록 프로그래밍 학습을 지원하는 인공지능 기반 교수학습 도구로서의 완성도를 갖추고자 하였다.

4.2.2 시스템 구현

본 연구에서 구현한 AI 에이전트는 Table 3과 같이 Next.js 프레임워크를 기반으로 구축되었으며, 데이터베이스는 Supabase, 배포 환경은 Vercel을 활용하였다.

Table 3. System overview

Category	Technical Name
Frontend	Tailwind CSS
Framework	Next.js
AI Model	OpenAI GPT-4o API
Database	Supabase
Deployment	Vercel

시스템 구현은 UI 구성, 데이터 처리 및 API 연동, 성능 최적화, 데이터베이스 연동의 네 단계로 진행하였다. 먼저, Fig. 4와 같이 EntryJS 작업 공간과 AI 에이전트 패널의 비율을 3:1로 나누어 화면을 구성하였다. Entry JS 작업 공간은 오픈소스(Apache License 2.0)를 기반으로 연구 목적에 맞게 수정하여 구현하였다[25].

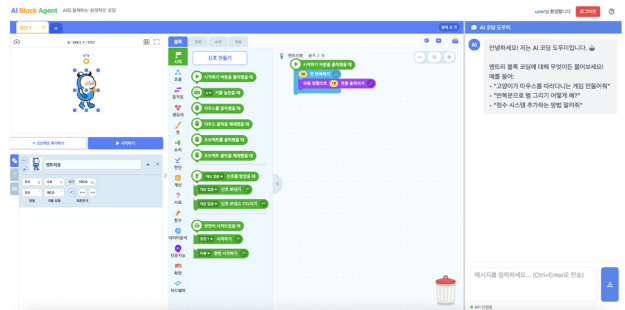


Figure 4. The user interface of the AIBlockAgent (The main screen showing the EntryJS workspace alongside the AI Agent panel)

다음으로 OpenAI의 GPT-4o API를 연동하여 자연어 질의 응답 기능을 구현하였다. 데이터 처리 흐름을 살펴보면, 학습자가 블록을 조립하는 순간 블록 조립소의 구조가 JSON 형태로 추출된다. ContextBuilder는 이 JSON 데이터에서 사용된 블록 ID를 파싱하고, BlockDictionary에서 관련 메타데이터를 검색한다. 이후 학습자의 현재 코드 상태, 대화 기록, 질문 내용 등을 하나로 통합한 프롬프트를 생성하여 AI 모델에 전달한다. 전체적인 데이터 처리 과정은 Fig. 5에 도식화하였다.

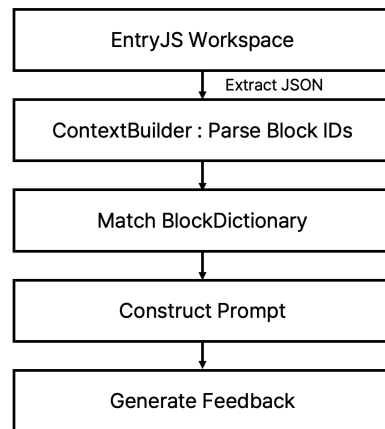


Figure 5. Data processing flow of AIBlockAgent

실시간 분석 과정에서 발생하는 서버 부하를 줄이기 위해 세 가지 최적화 기법을 적용하였다. 첫째, API 호출 빈도를 조절하였다. 학습자가 블록을 조립할 때마다 매번 분석을 요청하지 않고, 조작이 멈춘 뒤 일정 시간이 경과한 시점에만 분석을 수행하도록 설계하여 불필요한 서버 부하를 방지하였다. 둘째, 코드 상태 변화 감지이다. 이전 코드 상태와 현재 상태를 비교하여 실질적인 블록 구성의 변화가 확인된 경우에만 분석을 수행하도록 하였다. 셋째, 전송 데이터의 경량화이다. RAG 기반의 응답 생성을 통해 전체 블록 사전 데이터 중 현재 학습 상황과 관련이 있는 핵심 정보만을 선별하여 모델에 전달하였다. 이를 통해 토큰 사용량을 약 12% 절감하면서도 답변의 정확도를 유지할 수 있었다.

마지막으로 데이터베이스 연동을 통해 사용자 인증과 대화 로그 저장 기능을 추가하였다. 로그 데이터에는 사용자의 질문과 AI 응답뿐만 아니라 해당 시점의 블록 코드 정보를 함께 기록함으로써, 향후 학습자의 문제 해결 과정을 분석할 수 있는 기반을 마련하였다.

### 4.3 평가 및 반성

#### 4.3.1 평가 방법

평가 및 반성 단계에서는 AI 에이전트의 사용성을 검증하고자 전문가 평가를 실시하였다. 사용성 평가의 참가자는 선행 연구에 따라 10인으로 구성하였다. Faulkner(2003)에 따르면 5명의 사용자는 전체 문제의 85%를 발견할 수 있지만, 10명의 사용자부터는 평균 95% 이상의 문제를 찾아낼 수 있다[26]. 또한, Hwang & Salvendy(2010)는 여러 사용성 평가 연구를 분석하여 전체 문제의 80% 이상을 발견하기 위해서는 약 10±2명의 참여자가 필요하다는 규칙을 제시하였다[27]. 이에 본 연구에서는 Table 4와 같이 블록 프로그래밍 지도 경험이 풍부하고 컴퓨터교육 및 인공지능 교육 전문성을 갖춘 현직 교사 10인을 전문가 집단으로 선정하였다.

Table 4. Basic information of the expert group

Category	Description	Count
Highest Level of Education	Master's Course	4
	Master's Degree	5
	Doctoral Course	1
Field of Study	Computer Education	4
	AI Education	5
Teaching Experience	Other	1
	Less than 5 years	1
	5 to 10 years	4
	11 to 15 years	4
	more than 15 years	1
Block Programming Teaching Experience	Yes	10
	No	0

전문가들은 AI 에이전트를 직접 시연한 후 피드백 정확도, 유용성, 상호작용성 등의 측면에서 시스템을 평가하였다. 본 시스템이 블록 프로그래밍 학습을 지원하는 교육용 에이전트임을 고려하여, 평가 도구는 Holmes et al.(2023)이 개발한 챗봇 유용성 평가 문항을 연구 목적에 맞춰 수정 및 보완하여 활용하였다[28]. 기존의 '챗봇(chatbot)'이라는 용어를 'AI 에이전트(AI Agent)'로 변경하였으며, 특히 학습자와의 상호작용 과정에서 제공되는 피드백의 교육적 적절성을 확인할 수 있도록 문항을 조정하였다. 또한, 마지막에는 시스템 개선 및 추가 기능에 대한 자유 서술형 문항을 포함하여 정량적 결과를 보완할 수 있도록 구성하였다.

#### 4.3.2 1차 사용성 평가 결과

본 연구의 사용성 평가 문항은 총 16문항으로 구성되었으며, 각 문항은 5점 리커트(Likert) 척도로 응답하였다. 홀수 문항은 긍정적 진술, 짝수 문항은 부정적 진술로 구성되어 있다. 이에 짝수 문항의 경우, 일관된 분석을 위해 (6-응답값)으로 역코딩하였다. 분석의 목적은 총점 산출보다는 문항별 응답자의 인식을 통해 시스템의 세부적인 강점과 약점을 파악하는 데 두었다. 1차 사용성 평가의 문항별 평균, 표준편차, 중앙값은 Table 5에 제시하였다.

Table 5. Basic information of the expert group

Item	M	SD	Median
Q1	4.5	0.53	4.5
Q2	4.1	0.88	4
Q3	4.6	0.70	5
Q4	4.6	0.52	5
Q5	4.6	0.52	5
Q6	4.3	0.82	4.5
Q7	4.5	0.85	5
Q8	4.7	0.48	5
Q9	4.1	0.88	4
Q10	4.3	1.06	5
Q11	4	0.82	4
Q12	4.6	0.97	5
Q13	3.3	1.06	3.5
Q14	3.8	1.03	4
Q15	4.6	0.97	5
Q16	4.9	0.32	5

평가 결과, 전문가들은 AI 에이전트의 전반적인 사용성에 대해 대체로 긍정적으로 평가하였다. 특히 에이전트의 성격 및 태도 관련 문항(Q1, Q3, Q4)은 평균 4.5점 이상으로 높게 나타나 응답자들은 에이전트가 친근하고 긍정적인 상호작용을 제공한다고 보았다. 또한, 사용 편의성(Q7, Q8, Q15, Q16) 관련 문항에서도 평균 4.6점 이상을 기록하여 시스템 구조의 직관성을 확인하였다.

반면, 오류 대응 능력(Q13, Q14)은 각각 평균 3.3점과 3.8점에 머물렀다. 이는 에이전트가 엔트리 시스템의 동작 원리를 완벽하게 이해하지 못했기 때문이었다. 일부 응답자들은 AI 에이전트가 제시한 블록 조합이 사용자가 의도한 결과와 다르게 작동하였음을 지적하였다.

예를 들어, 사용자가 “키보드 방향키로 엔트리봇을 움직이게 하고 싶어.”라고 질문했을 때, AI 에이전트는 ‘위쪽 화살표 키를 눌렀을 때’, ‘방향을 0도 방향으로 정하기’, ‘이동 방향으로 10만큼 움직이기’ 블록을 사용하도록 안내하였다. 그러나 이를 실제로 조합하여 실행하였을 때, 엔트리봇 오브젝트가 오른쪽으로 이동하였다. 이는 엔트리봇 오브젝트의 기본 이동 방향이 90도로 설정되어 있어, ‘방향을 0도 방향으로 정하기’ 블록 대신 ‘이동 방향을 0도 방향으로 정하기’ 블록을 사용해야 하기 때문이다. 그럼에도 불구하고 AI 에이전트는 사용자에게 잘못된 블록을 지속적으로 제시하였다. 이는 AI 에이전트의 엔트리 오브젝트 속성과 블록에 대한 이해도 개선이 필요함을 시사한다.

이 외에도 자유 서술 의견을 범주화하여 도출한 핵심 개선 방향은 다음과 같다. 첫째, 시스템 정확도 개선이다. 응답자들은 AI 에이전트가 엔트리 시스템의 기본 작동 원리와 오브젝트 속성에 대한 이해가 부족함을 지적하였다. 또한, 엔트리 블록 간의 기능적 차이를 명확하게 구분하지 못해 잘못된 피드백을 제시하는 경우가 있어, 블록별 쓰임새와 실행 맥락에 대한 정확한 이해가 필요함을 언급하였다.

둘째, 학습자 수준에 따른 피드백 조정이다. 일부 응답자들은 에이전트의 피드백이 다소 길고 복잡하다는 점을 지적하였다. “AI가 한 번에 말을 너무 많이 해서 학생들에게는 어렵게 느껴질 수 있다.”, “과정을 순차적으로 나누어 설명해 주면 좋겠다.”와 같은 응답을 통해 초등학생의 인지 수준을 고려하여 긴 설명을 지양하고 단계별 핵심 내용 중심으로 피드백을 간소화할 것을 제안하였다.

셋째, 피드백의 정교화이다. AI 에이전트가 사용자의 블록 조립소 블록을 실시간으로 정확히 파악하지 못해 잘못된 피드백을 제공하는 사례가 있었다. 따라서, AI가 학습자의 블록 조립소 블록 상태를 실시간으로 정확히 추적해야 한다고 보았다. 또한 ‘계속 반복하기’와 ‘10번 반복하기’ 등 유사 블록 간의 선택 기준을 좀 더 구체적으로 피드백해야 한다는 의견도 있었다.

넷째, 사용자 인터페이스 개선이다. 일부 응답자는 현재 메시지 입력 후 전송하는 과정에서 번거로움을 지적하였다. Enter를 누르면 준비가 되고, Ctrl + Enter 또는 전송 버튼을 눌러야 질문이 전송되는 방식이 비효율적임을 지적하며 Enter 입력 시 바로 전송되는 절차로 개선할 필요가 있음을 제시하였다.

다섯째, 교육적 안전성 확보이다. 일부 응답자들은 AI 에이전트가 프로그래밍과 무관한 질문에 응답하는 사례를 지적하며, 교육 맥락과 무관한 질문에 대한 응답을 제한하는 프롬프트가 필요함을 언급하였다. “영화 추천해 줘”라는 질문에 초등학생에게 연령 제한이 있는 콘텐츠를 제시한 점을

근거로 AI 에이전트가 사용자 연령과 교육 목적을 인식하여 강력한 필터링 기능을 갖추어야 한다고 지적하였다.

### 4.3.3 프로토타입 개선

1차 사용성 평가 결과를 바탕으로 Fig. 6과 같이 프로토타입을 개선하였다. 개선은 응답의 정확도 향상, 피드백 구조 재설계, 블록 피드백 정교화, 사용자 인터페이스 개선, 교육적 안전성 확보의 다섯 가지 방향으로 이루어졌다.

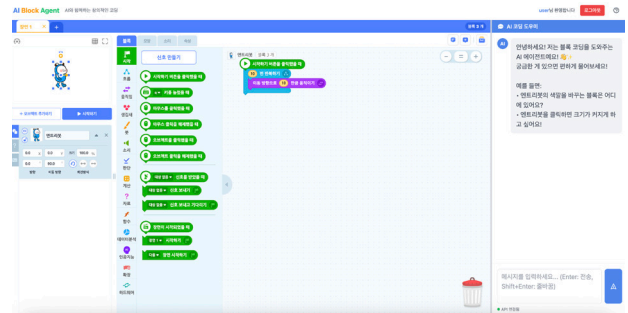


Figure 6. The user interface of the enhanced AIBlockAgent (Revised instructional prompts and functional adjustments in the AI Agent panel)

첫째, 시스템 정확도를 개선하였다. AI 에이전트가 엔트리 블록과 오브젝트의 속성을 명확하게 인식하도록 시스템 프롬프트를 고도화하였다. 이에 사용자가 “위쪽 화살표 키를 눌렀을 때 엔트리봇을 위로 움직이게 하고 싶어. 전체 블록을 알려줘.”라고 질문했을 때, 에이전트는 ‘위쪽 화살표 키를 눌렀을 때’ 블록과 ‘y 좌표를 10만큼 바꾸기’ 블록을 추천하였다. 다만, 동일한 질문을 반복했을 때, 에이전트가 ‘이동 방향으로 10만큼 움직이기’ 블록을 추천하는 경우도 있었다.

둘째, 학습자 수준에 따른 피드백 구조를 재설계하였다. 기존 프로토타입에서는 학습자가 시스템을 완성하기 위한 전체적인 절차를 한 번에 제시하여 정보량이 과도하다는 지적이 있었다. 이에 학습자의 단계별 성취에 맞춰 정보를 순차적으로 제공하도록 수정하였다. 이를 통해 초보 학습자가 느낄 수 있는 인지적 부담을 줄이고, 스스로 문제를 해결해 나가는 자기 주도적 학습 과정을 지원하고자 하였다.

셋째, 블록 피드백을 정교화하였다. AI 에이전트가 학습자의 블록 조립소 데이터를 실시간으로 수신하여 현재 코드 상태를 정확하게 분석할 수 있도록 기능을 강화하였다. 이를 통해 블록 누락, 잘못된 연결, 파라미터값 오류 등을 식별하여 현재 상태에 기반한 교정적 피드백을 제공하도록 하였다. 학습자의 코드 상태별 피드백 예시는 Fig. 7에 제시하였다.

넷째, 사용자 인터페이스를 개선하였다. 1차 평가에서 지적된 입력 방식의 번거로움을 해소하기 위해 Enter 키 입력으로 메시지가 전송되도록 변경하여 질의응답의 흐름을 개선하였다.

User's Block	Feedback
<p>시작하기 버튼을 클릭했을 때</p> <p>10 번 반복하기</p> <p>이동 방향으로 10 만큼 움직이기</p>	<p>내 이름을 묻고 인사하는 프로그램을 만들고 싶어.</p> <p>AI 이름을 묻고 인사하려면 먼저 사용자의 입력을 받을 수 있는 블록이 필요해요. '자료' 카테고리를 확인해보세요!</p>
<p>시작하기 버튼을 클릭했을 때</p> <p>안녕! 울(을) 울고 대답 기다리기 ?</p>	<p>이 블록이 맞는지 확인해줘</p> <p>AI 질문 블록은 잘 찾았어요! 😊 그런데 질문 내용이 "안녕!"이에요. 이걸 인사말이라고, 이름을 묻는 질문이 아니에요. '이름이 뭐예요?'처럼 질문을 바꿔보세요!</p>
<p>시작하기 버튼을 클릭했을 때</p> <p>이름이 뭐예요? 울(을) 울고 대답 기다리기 ?</p>	<p>그 다음은 어떻게 해?</p> <p>AI 이제 이름을 묻는 질문을 수정했으면, 입력 받은 이름으로 인사하는 블록을 추가해볼 차례예요. '계산' 카테고리에서 텍스트와 대답을 합치는 '합치기' 블록을 찾아보세요!</p>

Figure 7. Example of refined feedback (Providing corrective feedback based on the user's current block configuration)

다섯째, 응답 필터링을 강화하여 교육적 안전성을 확보하고자 하였다. AI 에이전트가 블록 프로그래밍과 직접적인 관련이 없는 질문에 응답하지 않도록 시스템 프롬프트를 보완하였다. 반면, 프로그래밍 관련 질문에 대해서는 학습자 수준에 맞춘 설명을 제공할 수 있도록 하였다. 이러한 응답 결과는 Fig. 8의 예시로 확인할 수 있다.

오늘 점심 뭐야?

AI 저는 엔트리 블록 코딩 도우미예요. 엔트리 블록과 관련된 내용을 물어봐 주세요!

변수가 뭐야?

AI 변수는 데이터를 저장할 수 있는 "상자"와 같아요. 프로그래밍에서는 정보를 담고, 필요할 때마다 값을 변경하거나 사용할 수 있도록 도와주는 역할을 해요. 엔트리 코딩에서 변수를 사용하려면 '자료' 카테고리에서 관련 블록을 찾아볼 수 있어요!

Figure 8. Example of the enhanced response filtering (The AI agent deflects an off-topic query while correctly answering an on-topic query)

#### 4.3.4 2차 사용성 평가 결과

2차 사용성 평가는 1차 평가에서 도출된 다섯 가지 개선 사항을 반영하여 보완한 프로토타입을 대상으로 실시하

였다. 2차 사용성 평가의 문항별 평균, 표준편차, 중앙값은 Table 6에 제시하였다.

Table 6. Results of the Second Usability Evaluation

Item	M	SD	Median
Q1	4.7	0.48	5
Q2	4.3	1.25	5
Q3	4.9	0.32	5
Q4	5.0	0.00	5
Q5	4.6	0.52	5
Q6	4.4	1.26	5
Q7	4.7	0.67	5
Q8	4.8	0.42	5
Q9	4.3	1.06	5
Q10	4.6	0.52	5
Q11	4.7	0.67	5
Q12	4.7	0.48	5
Q13	4.1	0.57	4
Q14	4.2	0.79	4
Q15	4.8	0.42	5
Q16	4.9	0.32	5

분석 결과, 모든 문항에서 평균 점수가 향상되었다. '에이전트는 모든 오류나 실수에 잘 대처하였다(Q13)' 항목은 평균이 3.3에서 4.1로 상승하였다. 이는 학습자의 실시간 코드 상태를 분석하고 검증하는 알고리즘을 정교화하여 응답의 정확도를 높인 결과로 볼 수 있다. 또한, '에이전트의 응답은 유용하고 적절하였다(Q11)' 항목의 평균도 4.0에서 4.7로 상승하여 피드백의 수준이 개선되었음을 확인하였다.

2차 사용성 평가를 통해 AIBlockAgent의 응답 정확도와 피드백 유용성 등 핵심 기능이 유의미하게 개선되었음을 확인하였다. 이는 교육설계연구의 순환적 설계 및 평가 절차가 시스템의 완성도를 높이는 데 기여하였음을 시사한다.

## 5. 결론

본 연구에서는 학교 현장에서 블록 프로그래밍 학습을 지원하기 위한 AIBlockAgent를 설계하고 개발하였다. 이를 위해 교사들이 현장에서 겪는 실제적인 어려움을 분석하고, 이를 해결하기 위한 네 가지 설계 원리를 도출하여 프로토타입을 제작하였다. 이후 전문가 집단을 대상으로 두 차례의 사용성 평가를 실시하여 시스템의 교육적 적절성과 유용성을 검증하였다.

연구 결과, AIBlockAgent는 에이전트의 태도, 상호작용성, 피드백의 유용성 등 전반에서 높은 사용성을 보였다. 학습자의 코드를 실시간으로 분석하고 개별화된 피드백을 제공한다는 점에서 교사 1인이 다수의 학생을 지도해야 하는 교실 환경을 보완하는 도구로서의 가능성을 확인하였다.

본 연구는 AIBlockAgent를 통해 학습자의 수준과 맥락에 적합한 개별화된 지원을 제공함으로써 맞춤형 프로그래밍 수업의 실천 가능성을 제시하였다. 그러나 다음과 같은 한계와 후속 연구 과제를 갖는다.

첫째, 현재 시스템은 엔트리의 데이터 분석, 인공지능 및 하드웨어 확장 블록을 지원하지 못해 활용 범위에 제약이 있다. 둘째, 일부 응답자는 태블릿 환경에서 화면 비율의 불편함을 지적하였다. 이에 모바일 환경에서의 UI 최적화가 요구된다.

셋째, 엔트리 블록 사전에 정의된 블록과 다른 명칭을 제시하는 오류가 여전히 존재한다. 본 연구에서는 시스템 프롬프트에 'Context에 제공된 블록 사건의 ID만 사용할 것', '블록 ID를 추측하거나 임의로 생성하지 말 것' 등의 제약 조건을 명시하고 있다. 그럼에도 AI 모델이 블록 사전에 없는 블록을 제안하는 사례가 확인되었다. 이는 모델이 사전 학습 과정에서 습득한 일반적인 프로그래밍 개념이 프롬프트 제약 조건보다 우선 적용된 것으로 보인다. 이를 해결하기 위해 후속 연구에서는 생성된 응답에 BlockDictionary에 정의된 블록만 포함되어 있는지 검증하는 방안을 모색할 필요가 있다.

넷째, 본 연구는 전문가 평가에 집중하여 실제 학습자를 대상으로 한 실증적 검증이 이루어지지 않았다. 이는 학습자 적용에 앞서 전문가 검토를 통해 교육적 적절성과 활용 가능성을 확인하고자 한 연구 설계에 따른 것이다. 그러나 전문가 평가와 실제 학습자 평가 사이에는 차이가 있을 수 있으므로, 후속 연구에서는 초보 학습자를 대상으로 AIBlockAgent를 적용하여 학습 효과와 사용성을 검증할 필요가 있다.

마지막으로, 최근 연구에 따르면 생성형 AI는 학습자의 창의적 사고를 촉진하는 동시에 과도한 의존을 초래하여 알고리즘적 사고의 발달을 저해할 우려가 있다[6, 29]. 따라서 향후 연구에서는 AI 에이전트와의 상호작용이 학습자의 문제 해결 과정과 인지적 변화에 미치는 영향을 종합적으로 분석할 필요가 있다.

## 참고문헌

- [1] Rwandan, S., Akram, B., Barnes, T., & Price, T. W. (2022). Adaptive immediate feedback for block-based programming: Design and evaluation. *IEEE Transactions on Learning Technologies*, 15(3), 406-420. <https://doi.org/10.1109/TLT.2022.3180984>
- [2] Shute, J. (2008). Focus on formative feedback. *Review of educational research*, 78(1), 153-189. <https://doi.org/10.3102/0034654307313795>
- [3] Zhou, Y., Pankiewicz, M., Paquette, L., & Baker, R. S. (2025). Impact of LLM Feedback on Learner Persistence in Programming. *Proceedings of the 33rd International Conference on Computers in Education*, Chennai, India. <https://library.apsce.net/index.php/ICCE/article/view/5957>
- [4] Kim, S. (2023). Developing Code Generation Prompts for Programming Education with Generative AI. *The Journal of Korean Association of Computer Education*, 26(5), 107-117. <https://doi.org/10.32431/kace.2023.26.5.009>
- [5] Choi, S., Lee, D., Kim, J., Jang, Y., & Kim, H. (2023). Designing LLM-based Code Reviewing Learning Environment for Programming Education. *The Journal of Korean Association of Computer Education*, 26(5), 1-11. <https://doi.org/10.32431/kace.2023.26.5.001>
- [6] Kazemitabaar, M., Chow, J., Ma, C. K. T., Ericson, B. J., Weintrop, D., & Grossman, T. (2023). Studying the effect of AI code generators on supporting novice learners in introductory programming. *Proceedings of the 2023 CHI conference on human factors in computing systems*, Hamburg, Germany, 1-23. <https://doi.org/10.1145/3544548.358091>
- [7] Kazemitabaar, M., Ye, R., Wang, X., Henley, A. Z., Denny, P., Craig, M., & Grossman, T. (2024). Codeaid: Evaluating a classroom deployment of an llm-based programming assistant that balances student and educator needs. *Proceedings of the 2024 chi conference on human factors in computing systems*, Honolulu HI, USA, 1-20. <https://doi.org/10.1145/3613904.3642773>
- [8] Xi, Z., Chen, W., Guo, X., He, W., Ding, Y., Hong, B., ... & Gui, T. (2025). The rise and potential of large language model based agents: A survey. *Science China Information Sciences*, 68(2), 121101. <https://doi.org/10.1007/s11432-024-4222-0>
- [9] Schlosser, M. (2019, December 21). Agency. *The Stanford Encyclopedia of Philosophy*. <https://plato.stanford.edu/archives/win2019/entries/agency/>
- [10] Zhang, X., Dong, X., Wang, Y., Zhang, D., & Cao, F. (2025). A Survey of Multi-AI Agent Collaboration: Theories, Technologies and Applications. *Proceedings of the 2nd Guangdong-Hong Kong-Macao Greater Bay Area International Conference on Digital Economy and Artificial Intelligence*, Dongguan, China, 1875-1881. <https://doi.org/10.1145/3745238.374553>
- [11] Deng, Z., Guo, Y., Han, C., Ma, W., Xiong, J., Wen, S., & Xiang, Y. (2025). AI agents under threat: A survey of key security challenges and future pathways. *ACM Computing Surveys*, 57(7), 1-36. <https://doi.org/10.1145/3716628>
- [12] Castelfranchi, C. (1994). Guarantees for autonomy in cognitive agent architecture. *International workshop on agent theories, architectures, and languages*, Berlin, Heidelberg, 56-70. [https://doi.org/10.1007/3-540-58855-8\\_3](https://doi.org/10.1007/3-540-58855-8_3)
- [13] Franceschelli, G., & Musolesi, M. (2025). On the creativity of large language models. *AI & society*, 40(5), 3785-3795. <https://doi.org/10.1007/s00146-024-02127-3>
- [14] Goodwin, R. (1995). Formalizing properties of agents. *Journal of Logic and Computation*, 5(6), 763-781. <https://doi.org/10.1093/logcom/5.6.763>
- [15] Wang, Z., Zhang, G., Yang, K., Shi, N., Zhou, W., Hao, S., ... & Fu, J. (2023). Interactive natural language processing. arXiv preprint arXiv:2305.13246. <https://doi.org/10.1145/3745238.374553>

- org/10.48550/arXiv.2305.13246
- [16] Lin, Y. T., & Chen, Y. N. (2023). Llm-eval: Unified multi-dimensional automatic evaluation for open-domain conversations with large language models. arXiv preprint arXiv:2305.13711. <https://doi.org/10.48550/arXiv.2305.13711>
- [17] Valový, M., & Buchalcevcova, A. (2023). The psychological effects of AI-assisted programming on students and professionals. *2023 IEEE International Conference on Software Maintenance and Evolution(ICSME)*, Bogotá, Colombia, 385-390. <https://doi.org/10.1109/ICSME58846.2023.00050>
- [18] Kohen-Vacs, D., Usher, M., & Jansen, M. (2025). Integrating Generative AI into Programming Education: Student Perceptions and the Challenge of Correcting AI Errors. *International Journal of Artificial Intelligence in Education*, 1-19. <https://doi.org/10.1007/s40593-025-00496-4>
- [19] Denny, P., MacNeil, S., Savelka, J., Porter, L., & Luxton-Reilly, A. (2024). Desirable characteristics for ai teaching assistants in programming education. *Proceedings of the 2024 on Innovation and Technology in Computer Science Education*, Milan, Italy, 1, 408-414. <https://doi.org/10.1145/3649217.3653574>
- [20] Ma, I., Krone-Martins, A., & Videira Lopes, C. (2024). Integrating ai tutors in a programming course. *Proceedings of the 2024 on ACM Virtual Global Computing Education Conference*, New York, USA 1, 130-136. <https://doi.org/10.1145/3649165.3690094>
- [21] Feng, T., Liu, S., & Ghosal, D. (2024). Courseassist: Pedagogically appropriate ai tutor for computer science education. *Proceedings of the 2024 on ACM Virtual Global Computing Education Conference*, 2, 310-311. <https://doi.org/10.1145/3649409.3691094>
- [22] McKenney, S., & Reeves, T. (2018). *Conducting educational design research*. Routledge. <https://doi.org/10.4324/9781315105642>
- [23] McKenney, S., & Reeves, T. C. (2025). Educational design research for relevant & robust scholarship. *Journal of Computing in Higher Education*, 1-25. <https://doi.org/10.1007/s12528-025-09456-2>
- [24] McKenney, S., & Reeves, T. C. (2021). Educational design research: Portraying, conducting, and enhancing productive scholarship. *Medical education*, 55(1), 82-92. <https://doi.org/10.1111/medu.14280>
- [25] Entry Labs. (n.d.). EntryJS [Computer software]. GitHub. <https://github.com/entrylabs/entryjs>
- [26] Faulkner, L. (2003). Beyond the five-user assumption: Benefits of increased sample sizes in usability testing. *Behavior Research Methods, Instruments, & Computers*, 35(3), 379-383. <https://doi.org/10.3758/BF03195514>
- [27] Hwang, W., & Salvendy, G. (2010). Number of people required for usability evaluation: the 10±2 rule. *Communications of the ACM*, 53(5), 130-133. <https://doi.org/10.1145/1735223.1735255>
- [28] Holmes, S., Bond, R., Moorhead, A., Zheng, J., Coates, V., & McTear, M. (2023, July). Towards validating a chatbot usability scale. *International Conference on Human-Computer Interaction*, Copenhagen, Denmark, 14033, 321-339. [https://doi.org/10.1007/978-3-031-35708-4\\_24](https://doi.org/10.1007/978-3-031-35708-4_24)
- [29] Mogavi, R. H., Deng, C., Kim, J. J., Zhou, P., Kwon, Y. D., Metwally, A. H. S., ... & Hui, P. (2024). ChatGPT in education: A blessing or a curse? A qualitative study exploring early adopters' utilization and perceptions. *Computers in Human Behavior: Artificial Humans*, 2(1), 100027. <https://doi.org/10.1016/j.chbah.2023.100027>



#### 최하은

- 2016년 한국교원대학교 초등교육과(학사)
- 2023년 서울교육대학교 교육전문대학원 인공지능 교육전공(석사)
- 2016년~2021년 서울덕의초등학교 교사
- 2022년~현재 서울난우초등학교 교사
- 2024년~현재 서울교육대학교 교육전문대학원 컴퓨터교육전공 박사과정

✚ 관심분야 : 양자 컴퓨팅 교육, 양자 알고리즘 교육, 데이터 과학

✉ choahaeun@gmail.com



#### 구덕희

- 2000년 한국교원대학교 초등교육과(박사)
- 2000년~2003년 한국교육학술정보원 선임연구원
- 2003년~2009년 대구교육대학교 컴퓨터교육과 교수
- 2009년~현재 서울교육대학교 정보교육과 교수

✚ 관심분야 : 컴퓨터교육, 데이터 과학, 디지털 리터러시

✉ dhk@snu.ac.kr