An Agile-Inspired Learner Model for Conversational AI in Education

Marina Buzzi[1][a], Agnese Camici[2], Barbara Leporini[2][b], and Angelica Lo Duca[1][c]
[1]*Institute of Informatics and Telematics, National Research Council, 56124, Pisa, Italy*
[2]*University of Pisa, 56127, Pisa, Italy*
{marina.buzzi, angelica.loduca}@iit.cnr.it, a.camici1@studenti.unipi.it, barbara.leporini@unipi.it

Keywords: Agile Methodology in Education, Artificial Intelligence, Educational Chatbots.

Abstract: Conversational agents are increasingly used in education, yet most educational chatbots remain content-centric and rely on learner models that represent knowledge states while leaving the learning process itself implicit. This paper introduces a process-oriented learner model that embeds principles from Agile methodology directly into the internal state of an AI-driven educational chatbot to support students' independent learning. In this model, learning objectives are represented as epics, study cycles as competence-driven sprints, and micro-level goals as user stories governed by explicit definitions of done. We implement the approach in a multi-agent Telegram chatbot used in an introductory university programming course. An exploratory feasibility study with 16 university students and recent graduates (N=16) provides preliminary evidence of the approach's pedagogical promise. Participants reported that the Agile-structured interaction improved goal clarity, step-by-step progression, and reflective engagement, though technical limitations in response latency and explanation depth were identified. This paper shows the feasibility of embedding explicit process models within conversational AI for education and motivates further investigation with larger samples and controlled evaluations.

1 Introduction

Artificial Intelligence in Education (AIED) increasingly explores technologies that support students' independent learning in flexible and context-sensitive ways (Simkins and Maier, 2023; Achuthan, 2025; Lan and Zhou, 2025; Wu et al., 2025). Conversational agents are well-suited to this model due to their immediacy and accessibility, and studies indicate that they can enhance engagement and provide on-demand assistance (Ait Baha et al., 2024; Kuhail et al., 2023). However, many educational chatbots still focus primarily on delivering explanations or answering isolated questions while offering limited support for structuring learners' progress toward learning goals over time (Debets et al., 2025; Guan et al., 2025). In parallel, Agile methodologies have been adopted in educational contexts primarily as course-level organizational frameworks, especially in project-based learning (Colissi et al., 2021; Mekić et al., 2024). While Agile concepts such as epics, sprints, and user stories have been used to structure courses, they are rarely operationalized as individual learning strategies and are seldom embedded computationally within AI-driven educational systems. At the same time, existing AI-based support for students' independent learning typically relies on abstract learner models, such as mastery levels or scores, without making the learning process itself explicit and checkable (Jin et al., 2023).

In this paper, we propose an Agile-aware educational chatbot that embeds Agile methodology directly into the learner model. Our approach builds on research in self-regulated learning (SRL), which identifies planning, monitoring, and reflection as key processes that learners must manage to study effectively (Zimmerman and Schunk, 2012). While traditional learner models primarily represent knowledge states (mastery learning or skill-based systems (Asher et al., 2025)) or model learner regulation processes (SRL scaffolding), the proposed model represents the structure of the learning process itself as an explicit, stateful workflow and uses that state to drive pedagogical decisions. While prior systems support self-regulated learning through prompts, dashboards, or feedback layers, the regulation process typically remains external to the learner model. In contrast,

[a] https://orcid.org/0000-0003-1725-9433
[b] https://orcid.org/0000-0003-2469-9648
[c] https://orcid.org/0000-0002-5252-6966

our approach encodes pedagogical progression (epics, sprints, user stories, and definitions of done) as computational state variables that directly drive dialogue orchestration and assessment. This shifts the learner model from representing what the learner knows to representing the learner's position within a structured learning process.

In our approach, Agile methodology is adopted as an explicit computational model of learner progression. Its value lies in providing a lightweight, interpretable process structure that can be directly encoded as learner state and used to govern conversational decisions. The learning objectives are represented as epics, study cycles as competence-driven sprints, and micro-level learning goals as user stories governed by explicit definitions of done. The chatbot serves as an Agile-aware learning scaffold, supporting planning, monitoring, feedback, and reflection during conversational interactions. We instantiate this approach in a multi-agent chatbot deployed in a university-level introductory programming course. An exploratory study with students suggests that the Agile-structured conversational interaction was generally perceived as clear, supportive, and helpful for engaging with course concepts, while also highlighting areas for improvement related to response time and explanation depth.

This paper makes two main contributions. First, we propose a conceptual model for representing the learning process itself as an explicit, computationally tractable Agile workflow embedded within a conversational AI system. This contrasts with traditional approaches that model learning outcomes (e.g., mastery levels, competence scores) but leave the process of reaching those outcomes implicit and unscaffolded. Second, we present preliminary empirical evidence from an exploratory deployment of a Telegram chatbot (N=16). This evidence shows that such process-oriented structuring is feasible to implement and is perceived by learners as supportive of goal clarity, progression awareness, and reflective engagement. While this early-stage evaluation does not establish learning effectiveness, it demonstrates proof-of-concept and identifies promising directions for future controlled studies.

2 Methodology

The proposed methodology adopts an Agile approach as a learner-centered strategy and integrates it directly into the internal state and interaction logic of a Multi-Agent-driven educational chatbot. In this approach, Agile is used as an explicit process model that structures how individual learners plan, execute, and reflect on their study activities. The chatbot maintains a persistent representation of this process for each learner and uses it to guide interaction, feedback, and progression in a competence-driven manner.

2.1 Agile Learning

At the core of the methodology is an Agile-native representation of the learner's learning process. High-level learning objectives from the course syllabus are represented as *epics* that define the overarching competence the learner aims to achieve. Each epic is addressed through one or more learning *sprints*, which correspond to focused study cycles centered on a single learning objective. Within each sprint, the learner progresses through a sequence of *user stories* that operationalize micro-level learning goals in concrete and testable terms. Progression through user stories is governed by an explicit definition of done that specifies the conditions under which a learning goal can be considered achieved, based on demonstrated understanding rather than time spent or task completion alone.

This structure supports key processes of self-regulated learning, including planning, monitoring, and reflection. Planning is supported by making epics and user stories explicit to the learner. Monitoring is achieved through continuous evaluation of learner responses against the definition of done for the active user story. Reflection is encouraged through feedback and retrospective prompts that invite learners to assess their own understanding and learning strategies. The chatbot serves as a scaffold, externalizing these processes and reducing the cognitive overhead of managing them independently.

2.2 The Multi-Agent Educational Chatbot

The model is operationalized as a multi-agent AI-based educational chatbot deployed on a Telegram messaging platform and centered on an Agile-aware orchestration layer (Figure 1). Learner messages are processed by an *Agile Workflow Orchestrator*, which maintains a persistent *Agile Learning State* for each student. This state represents the learner's progress in terms of Agile concepts, including the active learning epic, current sprint, active user story, and definition of done, thereby modeling the learning process rather than static knowledge levels.

Based on the current Agile Learning State, the orchestrator routes interactions to specialized agents. The *Assessment Agent* evaluates learner responses

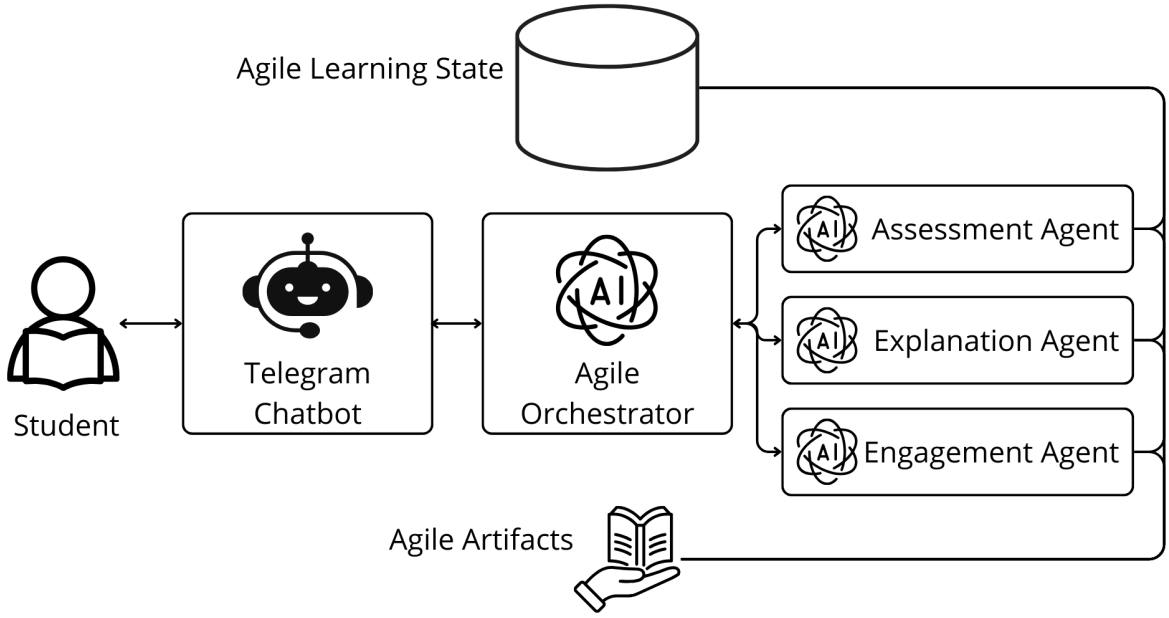

Figure 1: The architecture of the multi-agent educational chatbot.

against the definition of done for the active user story. The *Explanation Agent* provides targeted conceptual support when gaps or misconceptions are detected. The *Engagement Agent* supports focus, sprint continuity, and reflective prompts. All agent interactions are conditioned on the learner's Agile state to ensure alignment with the current learning sprint. Instructional content is organized as *Agile Learning Artifacts* bound to specific user stories and retrieved by the agents as needed. The orchestrator updates the Agile Learning State as user stories are completed or sprints conclude, enabling competence-driven progression. This architecture embeds Agile methodology directly into the learner model and interaction logic, supporting process-oriented and self-regulated learning through conversational AI.

2.3 Mapping Agile User Stories to Conversational Questions

The generation of conversational questions is driven by the learner's current Agile Learning State rather than by predefined question sets. At any point in the interaction, the chatbot maintains an explicit representation of the active epic, the current sprint, the active user story, and its associated definition of done. These elements jointly determine the instructional intent of the next system prompt. While the definition of done originates as a global acceptance criterion in software development, in this work it is adapted as a learning-level construct, specifying observable criteria for demonstrated understanding of individual learning goals.

To generate a question, the Agile Workflow Orchestrator conditions the language model on three components: the active user story, the criteria specified in the definition of done, and the relevant learning artifacts derived from the course slides. The resulting prompt instructs the model to generate a question that elicits evidence that the learner has satisfied the definition of done for the current user story. This ensures that questions are aligned with specific learning objectives and support competence-driven progression rather than generic practice. For example, when the active user story is "As a learner, I want to predict the output of a loop-based program," and the definition of done requires explaining loop execution step by step, the chatbot generates questions that present a concrete code snippet and explicitly ask the learner to reason about its behavior. If the learner's response does not meet the definition of done, the same information is used to generate targeted follow-up questions or explanations to close the identified gap.

2.4 An Example of an Application

The proposed approach is instantiated in the course Theoretical Foundations and Programming, taught at the university of Pisa. The course is structured around

two high-level learning objectives, represented in the chatbot as learning epics.

The first epic, *Theoretical Foundations*, aims to help students understand core theoretical concepts in computer science, including algorithms, computational procedures, and common misconceptions. The second epic, *Programming*, focuses on developing the ability to write, read, and reason about simple JavaScript programs, including control structures, functions, and basic data structures.

Each epic is addressed through a sequence of learning sprints that focus on specific portions of the course material as organized in the lecture slides. For example, within the Programming epic, a sprint may focus on iterative constructs. The sprint backlog is populated with user stories that operationalize this objective, such as explaining the purpose of loops, predicting the behavior of loop-based code examples presented in the slides, and modifying loop logic to achieve a different outcome. Each user story is associated with a definition of done specifying concrete criteria for demonstrated understanding.

During interaction, the chatbot retrieves slide-based learning artifacts associated with the active user story and uses them to generate context-specific questions that elicit evidence for the corresponding definition of done. Learner responses are then evaluated against these criteria, and targeted feedback is provided when needed. Progression through sprints is competence-driven, and once all user stories in a sprint are completed, the chatbot activates the next sprint or transitions to another epic, thereby structuring the study process as an explicit, interpretable Agile learning workflow.

Figure 2 illustrates the interaction flow of the agile-guided chatbot for the example described above. The sequence begins with the student initiating a session. The chatbot retrieves the current agile learning state from the Orchestrator, comprising the active epic, sprint, user story, and definition of done, and uses it to generate a context-specific question presented to the student. Upon receiving the student's answer, the chatbot forwards it to the Orchestrator, which delegates evaluation to the Assessment Agent. If the definition of done is not satisfied (Phase 3a), the Explanation Agent generates a targeted follow-up question, and the cycle repeats. Once the definition of done is met (Phase 3b), the Orchestrator updates the Agile Learning State and activates the next user story or, if the sprint backlog is complete, the next sprint. Progression is therefore entirely competence-driven rather than time- or interaction-count-based.

3 Exploratory Study

The system was evaluated in an exploratory study that assessed the feasibility, usability, and perceived educational value of an Agile-aware educational chatbot. The goal of this study was not to measure learning gains but to evaluate whether learners could meaningfully engage with an explicit process-oriented learner model instantiated through Agile constructs.

3.1 Setup

The evaluation was conducted in two phases. An initial pilot phase involved two university students enrolled in a course closely related to the target domain. The goal of this phase was to identify major usability and interaction issues before broader testing. During the pilot interactions, students completed a subset of learning activities using the chatbot and then filled in the structured questionnaire.

The questionnaire included Likert-scale items on conceptual clarity, theory–code connection, misconception handling, interaction comfort, and overall usefulness, as well as open-ended feedback. Based on these observations, the system was refined before the main evaluation. Response latency was reduced by optimizing agent orchestration and introducing immediate acknowledgment messages to signal ongoing processing. In addition, the coordination logic was improved to better maintain alignment between learner input, the active user story, and the current sprint, ensuring clearer progression through learning activities.

Analysis of pilot data and interaction logs revealed two main issues. First, students reported excessive response latency, which disrupted conversational flow and negatively affected engagement. Second, some interactions lacked clear continuity between questions, feedback, and subsequent prompts, occasionally leading to confusion about the current learning focus.

Following these refinements, the main evaluation involved 14 additional undergraduate students and recent graduates from a computer science–related program. Participants interacted with the chatbot during a single session lasting approximately 10 to 15 minutes and completed the full questionnaire afterward. The questionnaire included Likert-scale items and open-ended questions designed to capture perceived clarity of explanations, confidence in understanding, comfort during interaction, and overall usefulness of the chatbot as a study support tool.

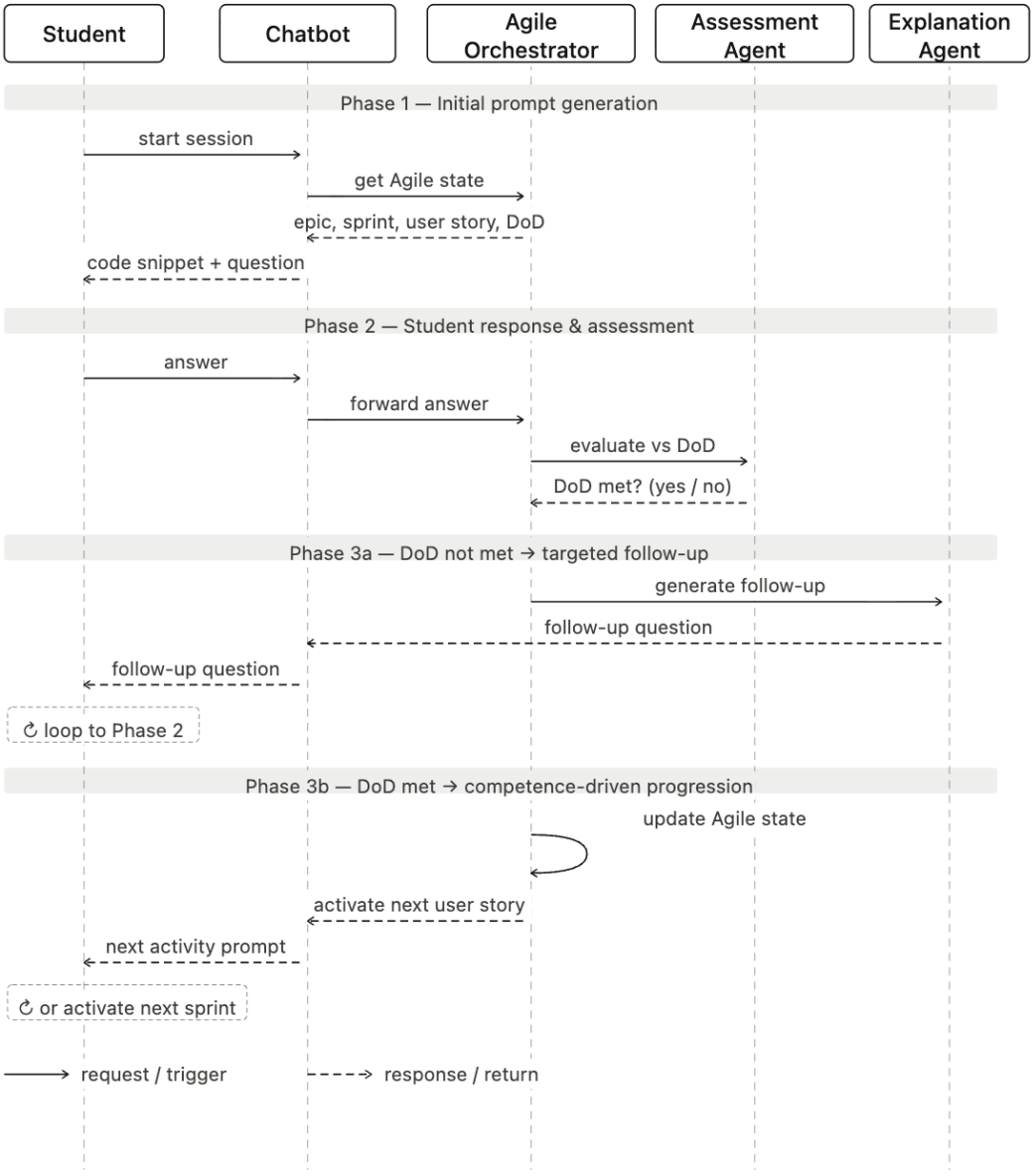

Figure 2: Sequence diagram of the Agile-guided interaction flow. Solid arrows indicate requests or triggers; dashed arrows indicate responses or return messages. Phase 3a (DoD not met) triggers a targeted follow-up cycle via the Explanation Agent; Phase 3b (DoD met) updates the Agile Learning State and activates the next user story or sprint.

3.2 Example of an Agile-Guided Interaction Flow

To concretely illustrate how the proposed Agile-oriented learner model drives dialogue, assessment, and progression, this section presents a simplified example from the Programming epic implemented in the chatbot. The chatbot maintains an explicit agile learning state for each learner. In this example, the active state is defined as follows:

- Epic: Programming
- Sprint: Iterative constructs
- User Story: As a learner, I want to predict the output of a simple loop-based program.
- Definition of Done: (i) correctly predict the final output of the program; (ii) explain step by step how the loop executes, including initialization, condition checks, and variable updates.

These elements are explicitly represented as state variables and constitute the primary control mechanism for interaction.

Initial prompt generation. Based on the active user story and its definition of done, the Agile Workflow Orchestrator instructs the system to generate a question that elicits evidence for the required understanding. The chatbot presents a short code snippet containing a loop and asks the learner to predict its output and explain the execution process.

Assessment and targeted follow-up. Suppose the learner correctly predicts the final output but provides an incomplete explanation of how the loop variable evolves across iterations. The Assessment Agent evaluates the response against the definition of done and determines that the user story is not yet completed, due to unmet explanation criteria. Instead of progressing to the next activity, the chatbot generates a targeted follow-up question focused specifically on the missing reasoning step (e.g., the condition evaluation or update rule), using the same Agile state information.

Completion and progression. Once the learner provides an explanation that satisfies all criteria in the definition of done, the user story is marked as completed. The Agile Workflow Orchestrator updates the learner's Agile Learning State and activates the next user story within the sprint or, if the sprint backlog is complete, initiates the next sprint. Progression is therefore competence-driven rather than time-based or interaction-count-based.

Unlike conventional educational chatbots that respond to isolated queries or rely on abstract mastery scores, the proposed system uses the learner's position within an explicit Agile workflow to govern dialogue, assessment, and progression. The learner model thus represents the learning process itself, and this process state directly determines system behavior at each interaction step.

3.3 Results

All students successfully completed the activities and the questionnaire. Self-reported prior knowledge of programming was heterogeneous (M = 3.25/5), while confidence with theoretical computer science concepts was lower on average (M = 2.69/5), indicating a cohort with mixed preparation and substantial room for conceptual support.

Overall, responses suggest that learners experienced the interaction as structured and progression-oriented. Participants rated the clarity of the bot's explanations highly (M = 4.50/5) and reported that the system explained code execution step by step (M = 4.31/5). Almost all students agreed that the bot's language was appropriate for their level of knowledge (M = 4.69/5), indicating that learners were able to follow the progression of activities.

Results also point to perceived support for monitoring understanding. Students reported that the bot helped them better understand a misconception (M = 4.06/5) and clearly explained why an answer was correct or incorrect (M = 4.00/5). More than half of the participants indicated that the interaction helped them grasp a concept that had previously been unclear (M = 4.06/5). These responses suggest that the system's use of explicit completion criteria and targeted follow-up questions may have supported learners in evaluating the adequacy of their understanding during the interaction.

The interaction was further associated with increased confidence. Participants reported greater confidence in their understanding of what an algorithm is (M = 4.38/5) and greater ability to modify or reuse the provided code example (M = 4.56/5). Although self-reported, these results indicate that learners perceived the interaction as enabling more independent reasoning about code and underlying concepts.

The affective climate was consistently positive: almost all students felt comfortable asking questions (M = 4.94/5), most indicated that the bot's tone did not make them feel judged (M = 4.63/5), and trust in the bot's explanations increased after the interaction (M = 4.19/5). Such perceptions are important for process-oriented support, as they create conditions that increase the likelihood that learners will engage in iterative questioning and reflection.

Finally, most participants indicated that they would consider using the bot as a study support tool

(8 “yes,” 3 “maybe”, 3 “no”), particularly in combination with slides or textbooks, and for exam revision. This suggests that learners perceived the chatbot not as a replacement for existing materials but as a structured companion that could guide their study process.

Conversation analysis indicates that the system sometimes repeated the same examples, and users found the automatic follow-up questions at the end of responses not always useful, suggesting that greater user control could improve clarity and efficiency.

3.4 Limitations

This exploratory study has several limitations. First, it involved a small pilot sample (N=16), limiting generalizability. Second, the evaluation measured learners' perceptions of clarity, usefulness, and comfort but did not assess objective learning gains; future studies should employ pretest-posttest- or comparative designs. Third, the study took place in a single university course on theoretical foundations and programming, using slide-based materials, which may limit transferability to other contexts. Fourth, only short interaction sessions were examined; therefore, the longer-term appropriation of the Agile learning structure and its effects on self-regulated learning remain unknown. Fifth, the prototype showed technical constraints, including response latency and occasional lack of depth, which may have influenced user experience.

In addition, a structured Agile progression may not suit all learners equally, and differences in learning preferences were not investigated.

Finally, the system operates on a pervasive messaging platform (Telegram), raising ethical and privacy concerns, including data sensitivity and user consent. In this exploratory phase, the study focused on feasibility and usability and did not collect sensitive personal data beyond learning interaction logs. All collected data were used solely for research purposes. These issues will be addressed more systematically in future work.

4 Conclusions and Future Work

This paper introduced an Agile-aware educational chatbot that embeds Agile methodology directly into the learner model to support process-oriented and independent learning. The proposed approach represents the learning process as an explicit, computationally tractable workflow that structures planning, monitoring, and reflection during interaction.

The process-oriented control mechanism contrasts with conventional educational chatbots, where adaptation typically occurs at the level of content selection or response generation, while the structure of the learning process remains implicit or externally managed by the learner.

The exploratory study suggests that students perceived this process-oriented conversational support as clear, structured, and helpful for maintaining focus and progression, providing initial evidence of the feasibility of learners engaging with an explicit pedagogical process model instantiated through conversational AI. More broadly, this work highlights the potential of process-oriented learner models as a complementary direction to traditional knowledge-based approaches in AIED, positioning pedagogical progression as a first-class element of computational learner modeling.

Overall, the contribution of this work should be interpreted primarily as architectural and methodological: it demonstrates the feasibility of embedding an explicit process model within conversational AI, rather than providing evidence of learning effectiveness.

Future work will pursue three directions: (1) larger-scale, longitudinal studies to examine how sustained use of a process-oriented learner model affects learning outcomes and self-regulated learning; (2) technical improvements to enhance robustness, including latency, explanation depth, and interaction reliability; and (3) adaptive mechanisms for prioritizing epics, sprints, and user stories, so the Agile structure can better align with individual learner needs and evolving course constraints.

REFERENCES

Achuthan, K. (2025). Artificial intelligence and learner autonomy: a meta-analysis of self-regulated and self-directed learning. In *Frontiers in Education*, volume 10, page 1738751. Frontiers Media SA.

Ait Baha, T., El Hajji, M., Es-Saady, Y., and Fadili, H. (2024). The impact of educational chatbot on student learning experience. *Education and Information Technologies*, 29(8):10153–10176.

Asher, M. W., Hartman, J. D., Blaser, M., Eichler, J. F., and Carvalho, P. F. (2025). The promise of mastery-based testing for promoting student engagement, self-regulated learning, and performance in gateway stem courses. *Computers & Education*, page 105387.

Colissi, M. d. S., Vieira, R., Mascardi, V., and Bordini, R. H. (2021). A chatbot that uses a

multi-agent organization to support collaborative learning. In *Proceedings of HCI International*, Lecture Notes in Computer Science, pages 31–38. Springer.

Debets, T., Banihashem, S. K., Joosten-Ten Brinke, D., Vos, T. E., Maillette de Buy Wenniger, G., and Camp, G. (2025). Chatbots in education: A systematic review of objectives, underlying technology and theory, evaluation criteria, and impacts. *Computers Education*, 234:105323.

Guan, R., Raković, M., Chen, G., and Gašević, D. (2025). How educational chatbots support self-regulated learning? a systematic review of the literature. *Education and Information Technologies*, 30(4):4493–4518.

Jin, S.-H., Im, K., Yoo, M., Roll, I., and Seo, K. (2023). Supporting students' self-regulated learning in online learning using artificial intelligence applications. *International Journal of Educational Technology in Higher Education*, 20(1):37.

Kuhail, M. A., Alturki, N., Alramlawi, S., and Alhejori, K. (2023). Interacting with educational chatbots: A systematic review. *Education and Information Technologies*, 28(1):973–1018.

Lan, M. and Zhou, X. (2025). A qualitative systematic review on ai empowered self-regulated learning in higher education. *npj Science of Learning*, 10(1):21.

Mekić, E. S., Jovanović, M. N., Kuk, K. V., Prlinčević, B. P., and Savić, A. M. (2024). Enhancing educational efficiency: Generative ai chatbots and devops in education 4.0. *Computer Applications in Engineering Education*, 32(6):e22804.

Simkins, S. and Maier, M. (2023). *Just in time teaching: Across the disciplines, and across the academy*. Taylor & Francis.

Wu, X.-Y., Radloff, J. D., Yeter, I. H., Wang, L., and Chiu, T. K. (2025). Designing artificial intelligence chatbots for self-regulated learning from a systematic review based on habermas's three interests. *Interactive Learning Environments*, pages 1–24.

Zimmerman, B. J. and Schunk, D. H. (2012). Motivation: An essential dimension of self-regulated learning. In *Motivation and self-regulated learning*, pages 1–30. Routledge.