Koh, E., et al. (Eds.) (2026). Proceedings of the 2nd International Conference on Learning Evidence and Analytics. Asia-Pacific Society for Computers in Education.

Orchestrated Pedagogical Agents for Efficient Language Learning: An MCP-Enabled Meta-Agent Framework

Isanka WIJERATHNE[a*], Chia-Yu HSU[a], Steve WOOLLASTON[b], Brendan FLANAGAN[c] & Hiroaki OGATA[a]

[a]*Academic Center for Computing and Media Studies, Kyoto University, Japan*
[b]*Graduate School of Informatics, Kyoto University, Japan*
[c]*Center for Innovative Research and Education in Data Science, Kyoto University, Japan*
*wijerathne.isanka.6z@kyoto-u.ac.jp

Abstract: Large language model (LLM)-backed tutors can provide personalized explanations and conversational practice, but current deployments often take one of two suboptimal forms: a monolithic general-purpose agent with inconsistent pedagogy and shallow scaffolding, or a fragmented multi-bot ecosystem that imposes cognitive overhead by forcing learners to switch between specialist chatbots and disrupting learning flow. We introduce OPA (Orchestrated Pedagogical Agents), a meta-agent framework that coordinates task-specific tutors through an explicit pedagogical policy grounded in learning science. OPA leverages the Model Context Protocol (MCP) to standardize access to learner state, practice banks, content repositories, and logging, enabling modular design and reproducible experimentation. OPA implements a progressive tutoring protocol emphasizing minimal effective help, diagnostic learner production, adaptive micro-practice, and scaffolding fading. We outline a planned three-condition evaluation (monolithic, manual multi-bot, and OPA) measuring learning efficiency (gain/min), transfer, delayed retention, and learner experience. This paper presents OPA's architecture and protocol as a design contribution; empirical validation is reserved for future work.

Keywords: LLM tutoring, pedagogical orchestration, meta-agent, adaptive practice, learning efficiency, Model Context Protocol (MCP), multi-agent tutoring

1. Introduction

Language learning requires sustained practice and timely feedback across micro-tasks such as reading, grammar, vocabulary, writing, and role-play. Because learner needs shift within and across sessions, effective tutors must adapt scaffolding, select appropriate interventions, and sequence practice based on task context and learner state.

Recent LLM-based tutors provide fluent explanations and feedback. However, large-scale LLM tutoring deployments show both promise and risks. **Monolithic tutors** rely on a single general-purpose agent, which can yield inconsistent pedagogy and shallow scaffolding across task types. For example, 61A-Bot reduced homework completion time at scale but reported mixed downstream effects and concerns about overreliance (Zamfirescu-Pereira et al., 2025). Tutorly transformed learning materials into interactive tutoring with measurable pre/post gains, highlighting the importance of instructional structure beyond fluent feedback (Li et al., 2024). These findings motivate policies that balance assistance with productive struggle.

Multi-agent educational systems have emerged as an alternative to monolithic tutoring. MAIC proposed LLM-driven multi-agent "classrooms" with pilot evidence (Yu et al., 2024). **Pair-Up** explored human-AI co-orchestration of transitions between individual and collaborative learning, highlighting tensions around who controls orchestration decisions (Yang et al., 2023). **Manual multi-bot** ecosystems provide specialized chatbots (Woollaston

et al., 2025) for activities such as grammar correction or role-play, but push coordination to learners, disrupting learning flow and adding switching overhead (Li et al., 2024).

This paper introduces Orchestrated Pedagogical Agents (OPA), a meta-agent framework that coordinates specialized tutoring bots through an explicit pedagogical policy. Unlike simple intent-classification routing, OPA makes pedagogical decisions at each learner turn, selecting specialist capabilities, calibrating scaffolding levels, assigning targeted micro-practice, and enforcing productive struggle through reattempt loops. The framework leverages the **Model Context Protocol** (MCP) to standardize access to learner models, practice banks, content repositories, and logging infrastructure, enabling modular design, experimental reproducibility, and operational auditability. For scaffolding and control, recent approaches combine symbolic pedagogical rules with LLM reasoning (Tong & Hu, 2024), improve feedback clarity via response rephrasing (Lin et al., 2025), and analyze dialog moves linked to learning gains while identifying counterproductive patterns (Borchers et al., 2024). OPA builds on these directions by (i) treating orchestration as a pedagogical policy decision, and (ii) using MCP to standardize interfaces to learner modeling, practice, content, and logging for reproducible experimentation.

2. System Overview of Orchestrated Pedagogical Agents (OPA)

2.1 OPA Framework Concept

OPA converts fragmented specialist tutoring bots (e.g**., Kōrero** EFL apps such as **TAMMY**, **ARCHIE**, and **Penny**) into a unified learning interface. A **Meta-Agent Orchestrator** selects specialist capabilities and follow-up actions based on task context, learner state, and struggle signals, while enforcing pedagogical constraints (e.g., learner production and no direct answers in assessed tasks). Each capability is exposed as an MCP server, enabling modular integration and controlled experimental ablations. This architecture replaces the manual multi-bot navigation burden placed on learners and reduces coordination overhead, while supporting reproducible research across alternative orchestration strategies.

2.2 MCP Integration

MCP standardizes how an LLM application accesses external tools and context using a host-client-server pattern with a stateful JSON-RPC session protocol for context exchange and coordinated sampling. Servers expose (i) tools (structured functions), (ii) resources (data via URIs), and (iii) prompts (versioned templates). In OPA, the orchestrator acts as the MCP client and discovers and invokes servers for the learner model, practice bank, content repository, logging/analytics, specialist prompts, and the pedagogical policy layer. This separation enforces clear security boundaries and separation of concerns, while supporting traceability and controlled component swapping for ablations.

2.3 Architecture

The system architecture (Figure 1) shows the student interacting with a unified tutor interface, which connects to the Meta-Agent Orchestrator. The orchestrator interacts with several MCP servers: **Pedagogical Policy Layer** implements decision logic for routing, scaffolding calibration, and practice assignment. **Learner Model Server** exposes learner state (proficiency estimates, mastery signals, recent errors) and update tools. **Practice Bank Server** provides targeted micro-practice recommendations. **Content Server** delivers passages, curriculum materials, and teacher-provided texts. **Specialist Prompt Server** maintains versioned prompt families for each specialist capability. **Logging/Analytics Server** records events and provides summary resources. **Safety & Guardrails** enforce pedagogical constraints. All servers connect to a shared LLM runtime for generation, enabling controlled ablation studies by swapping server implementations or prompt versions.

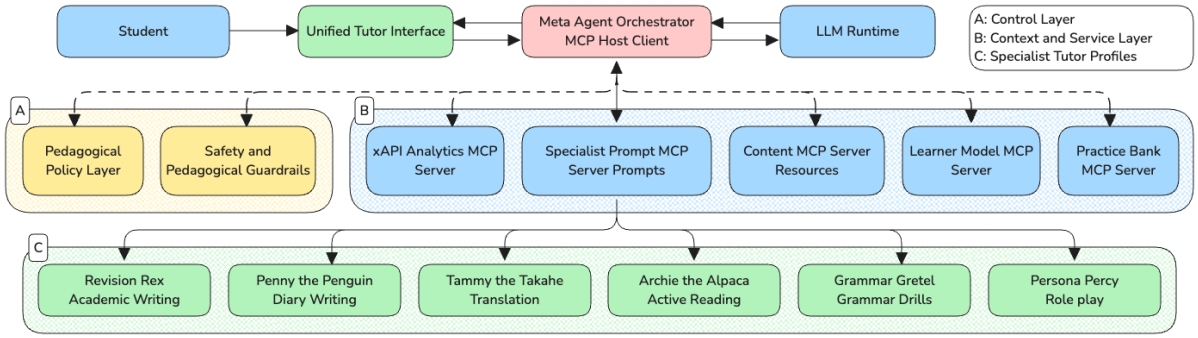

Figure 1: OPA Architecture with MCP-Based Capability Servers.

3. Pedagogical Orchestration Framework with OPA

3.1 Orchestration as a Pedagogical Decision Problem

OPA treats orchestration as a **pedagogical decision problem**, not simple intent routing. At each learner turn, the policy selects an instructional move that increases the likelihood of learning under time and engagement constraints. Policy inputs summarize **task context** (current prompt/text and recent dialog), **learner state** (estimated proficiency and skill-tag mastery), **struggle/engagement signals** (repeated errors, retries, long pauses, confusion markers), and **pedagogical constraints** (e.g., avoid full solutions in assessed tasks; enforce learner production and reattempt loops). The policy outputs a structured tutoring plan specifying the **specialist capability**, **scaffolding level, target skill tags**, and the required **next learner action** (e.g., revise, answer, explain, continue).

3.2 Progressive Tutoring Protocol

OPA operationalizes minimal effective help through adaptive escalation. It starts with a minimal hint, then requires diagnostic learner production (e.g., explain the issue or rewrite) to confirm understanding. If errors persist, OPA assigns a short micro-practice loop (typically 3-5 items) targeting the limiting skill, followed by a reattempt of the original task with reduced support. Over time, OPA applies fading by lowering scaffolding as mastery improves. This protocol aligns with prior work on effective dialog moves and controlled scaffolding in LLM-mediated tutoring (Borchers et al., 2024; Lin et al., 2025; Tong & Hu, 2024) and directly addresses overhelping risks noted in recent deployments (Zamfirescu-Pereira et al., 2025).

3.3 MCP Servers as Pedagogical Infrastructure

OPA implements core pedagogical services as MCP servers. The Learner Model server exposes learner state (profile, mastery estimates, recent errors) and update tools. The Practice Bank server recommends short skill-tagged items and provides micro-practice templates. Content and Logging servers provide passages/curriculum targets and record events for teacher audit and research analysis. Specialist behaviors (e.g., grammar coach, translation feedback, role-play partner) are maintained as versioned prompt families to support reproducibility and controlled ablations.

4. Conclusion and Future Work

This paper presented OPA, an MCP-enabled meta-agent framework for pedagogical orchestration. OPA addresses a key gap in educational AI by moving beyond conversational fluency toward pedagogically sound instructional sequencing, offering a path to scalable multi-bot tutoring ecosystems that improve learning efficiency while maintaining learner engagement. This initial study focuses on EFL learners and text-based interactions, but the

framework can be extended to other domains, multimodal interaction (e.g., speech and images), and longer-term deployments through its modular integration of domain-specific specialist bots and practice banks as MCP servers.

As the future work, we are planning a three-condition randomized controlled study comparing (A) a monolithic general-purpose tutor, (B) a manual multi-bot condition where learners choose among specialist bots, and (C) OPA orchestration through a single unified interface. We plan to recruit up to 180 EFL learners (CEFR A2-B2) for a four-week intervention with a delayed post-test. Outcomes will include standardized pre/post/delayed EFL measures, learning efficiency (gain per minute), transfer tasks using novel prompts, and learner experience ratings; for OPA we will additionally log intervention/scaffolding distributions and conduct a small teacher audit sample to check overhelping and misrouting. Specifically, we will clarify:

- Whether pedagogical orchestration improves learning gains compared to monolithic tutors and manual multi-bot selection (Learning outcomes)
- Whether orchestration improves transfer to novel prompts and delayed retention (Transfer and retention)
- Whether orchestration improves learning gain per minute (Learning efficiency)
- Which policy components contribute most to learning outcomes (Policy components)

The above is expected to provide empirical evidence that pedagogically informed orchestration improves learning efficiency and outcomes beyond monolithic or manually coordinated multi-bot systems, while introducing a progressive tutoring protocol for LLM-based tutors that operationalizes minimal effective help, diagnostic production, adaptive practice, and scaffolding fading. It will also deliver a modular, reproducible MCP-based platform that standardizes capability interfaces, enabling researchers to replicate, compare, and extend orchestration policies, and generate design insights into which policy components (e.g., routing, scaffolding calibration, micro-practice, and fading) contribute most to learning gains.

Acknowledgements

This work is supported by CSTI SIP Grant Number JPJ012347 and JSPS KAKENHI Grant Number 23H00505.

References

Borchers, C., Yang, K., Lin, J., Rummel, N., Koedinger, K. R., & Aleven, V. (2024). Combining dialog acts and skill modeling: What chat interactions enhance learning rates during ai-supported peer tutoring? In *Proceedings of the 17th International Conference on Educational Data Mining* (pp. 117-130).

Lin, J., Han, Z., Thomas, D. R., Gurung, A., Gupta, S., Aleven, V., & Koedinger, K. R. (2025). How can i get it right? using gpt to rephrase incorrect trainee responses. *International journal of artificial intelligence in education*, *35*(2), 482-508.

Tong, R., & Hu, X. (2024). Future of Education with Neuro-Symbolic AI Agents in Self-Improving Adaptive Instructional Systems. Frontiers of Digital Education, *1* (2), 198-212.

Woollaston, S., Flanagan, B., & Ogata, H. (2025). Chatbot Personalisation for EFL Learning: Integrating BKT and Task Context. International Conference on Computers in Education

Yang, K. B., Echeverria, V., Lu, Z., Mao, H., Holstein, K., Rummel, N., & Aleven, V. (2023). Pair-up: prototyping human-AI co-orchestration of dynamic transitions between individual and collaborative learning in the classroom. Proceedings of the 2023 CHI conference on human factors in computing systems.

Zamfirescu-Pereira, J., Qi, L., Hartmann, B., DeNero, J., & Norouzi, N. (2025). 61A Bot Report: AI Assistants in CS1 Save Students Homework Time and Reduce Demands on Staff.(Now What?). Proceedings of the 56th ACM Technical Symposium on Computer Science Education V. 1.

Li, W., Pea, R., Haber, N., & Subramonyam, H. (2024). Tutorly: Turning Programming Videos Into Apprenticeship Learning Environments with LLMs. *arXiv:2405.12946.*

Yu, J*., et al. (2024).* From MOOC to MAIC: Reshaping Online Teaching and Learning through LLM-driven Agents. *arXiv:2409.03512.*