Reflection Report

Collaborative Human-LLM Interaction in Software Engineering

El Heyba EL HEYBA
Github: ALMA Project

February 8, 2026

1 The Learning Scenario

1.1 Context: Programming

The selected learning scenario is situated within a high-level collaborative environment—such as a specialized software engineering team or a research group—composed of multiple professionals with LLMs.

To maximize the collective intelligence of such a group, it is necessary to optimize the fundamental "atomic unit" of collaboration: the dyad composed of **One Professional paired with One Personal LLM**.

The hypothesis of this project is that the quality of the larger group's output is strictly limited by the quality of collaboration within these individual dyads. If the individual professional fails to engage deeply with their AI partner, the collective output of the group suffers.

Uniquely, this scenario identifies **two distinct learners** within the dyad, creating a reciprocal learning loop:

- **Learner A: The Human Professional (or Advanced Student):** An expert in the broad domain (e.g., Computer Science) but a "learner" in the specific context of the new problem (e.g., implementing a specific novel architecture) and in the skill of expressing needs to the LLM. Their goal is to refine their mental models and verify their strategic choices.
- **Learner B: The LLM:** Contrary to the standard view of AI as a static repository, in this collaborative model, the LLM is also a learner. It must "learn" the specific constraints, style, and intent of the professional. As noted in the system design, the AI learns to adapt its behavior based on the professional's level of engagement, effectively fine-tuning its persona to the human's needs, knowing when to provide assistance and when to withhold it.

Why is collaboration with an LLM relevant in this context?

As highlighted in the project context, a critical failure mode occurs when humans treat the LLM as an Oracle rather than a partner. In a group setting, this leads to two specific risks:

- **Homogenization of Solutions:** If every professional in the group merely accepts their LLM's initial response, the diversity of unique human perspective collapses. True creativity emerges only when individuals force the AI to adapt to specific constraints; without this friction, the group risks converging on identical, generic solutions.
- **The "Smartest Student" Fallacy:** If the LLM provides direct answers, it effectively becomes the "smartest student" in the room, reducing human professionals to passive consumers. This creates a dependency where humans adopt solutions they do not fully understand, preventing them from contributing original value or critical defense of the work during group discussions.

Pedagogical Goal: Metacognition through Verbalization

The objective of this system is to shift the Human-LLM dynamic from **Dependency** (Master-Slave) to **Interdependency** (Partner-Partner).

- **Verbalization as Learning:** The system compels the professional to articulate their logic before receiving assistance. This leverages the *Protégé Effect*, grounded in the principle that the act of defending an idea clarifies the learner's own understanding. This requirement ensures the human possesses a robust mental model before bringing their concepts to the larger group.
- **Preserving Human Agency:** To resolve the authority conflicts typical of collaborative learning, the professional retains strict decision-making power. The LLM's role is to **critique, optimize, and accelerate**, but never to dictate.
  - **No Assumption of Intent:** The AI is prohibited from making decisions on the professional's behalf or assuming ambiguous intent.
  - **Neutral Decision Support:** When asked to make a decision, the LLM instead inquires about the professional's criteria to facilitate *their* choice. When providing suggestions or alternatives, it presents them objectively—without influence or praise—ensuring the final strategic decision remains exclusively human.

2 View of Collaboration: Theoretical Framework

2.1 Defining Collaborative Learning in this Scenario

In this context, collaborative learning is defined as a process of **Mutual Regulation** and **Co-Construction**. It is not merely the exchange of information, but the joint creation of a solution where the outcome transcends what either the human or the LLM could achieve individually.

Unlike a standard "user-tool" interaction, here the learning is bidirectional: the Human Professional learns architectural rigor, while the LLM "learns" the specific constraints and intent of the project through the human's verbalization.

A central tenet of this system is that collaboration requires *effort*. If an AI partner eliminates cognitive effort by providing immediate solutions, it ceases to be a collaborator and becomes a crutch.

To prevent the "Free Rider" effect—where the human delegates all cognitive load to the machine—the system implements a **Constructive Friction** protocol.

- **Conditional Generation:** The system is programmed to withhold code generation until the user has demonstrated a clear conceptual understanding. If the input is vague or "lazy" (e.g., *"Write a script for X"*), the model refuses to execute and instead pivots to a reflective question (e.g., *"What is your strategy for handling X?"*).
- **The "Lead" Principle:** As established in the scenario analysis, the professional must always have the "last word." The LLM provides information, optimization strategies, and critiques, but it never makes the final decision. This ensures that the human remains the architect of the solution, while the AI acts as the accelerator.

2.2 Adaptive Collaboration: The "Expertise" Variable

Collaboration in this system is not static; it is a dynamic, reciprocal process. The prototype introduces a `collaboration_level` that adjusts in real-time based on the professional's input:

- **Reciprocity of Effort:** The system follows a "Mirroring Principle." If the professional provides detailed context and thoughtful constraints, the LLM reciprocates with deeper, more complex ar-

chitectural critiques. The more the professional collaborates, the more value they unlock from the partner.

- **The "Explicit Ignorance" Trigger:** By default, the LLM assumes the user is competent and maintains a strict "Critic" persona (challenging the user). It will *only* shift to a "Scaffolding/Mentor" persona (explaining concepts or giving examples) if the professional **explicitly states they do not know** a specific concept. This prevents the AI from "over-explaining" to an expert while ensuring help is available when genuinely requested. Even in this case, the LLM provides the required knowledge to the professional so that he will be able to solve the task himself.

3 Critical Reflection and Future Work

3.1 Design Rationale

Why this Scenario? I selected High-Level Software Architecture because it is a domain where "correctness" is subjective. Unlike simple syntax errors, architectural choices require justification, making them the ideal ground for testing whether a user can defend their ideas against an AI critic.

Why this Behavior? The choice to implement "Constructive Friction" was driven by the observation that efficiency is often the enemy of learning. By forcing the user to pause and negotiate, we trade short-term speed for long-term metacognitive gain.

3.2 Limitations and Trade-offs

While the approach yields higher educational value, it introduces significant technical and pedagogical risks:

- **Constraint Overload and Performance:** Imposing too many behavioral constraints (Neutrality, Gatekeeping, Critique) consumes the LLM's "context window" and reasoning capacity. This can lead to a decrease in the quality of the technical solution or increased latency. To maintain high performance under these heavy constraints, the system may require a higher-tier model (e.g., Gemini Ultra or GPT-4) rather than lighter, faster models.
- **The "Pedagogical Deadlock":** A critical failure mode exists where the professional genuinely cannot solve the problem, even with scaffolding, but the LLM's "Anti-Free-Rider" protocol refuses to provide the solution. This results in a deadlock where the user is stuck and frustrated, rather than learning. Future versions must implement a "surrender" mechanism where the LLM can detect genuine exhaustion and provide the solution as a last resort.

3.3 Future Improvements and Evaluation

If more time were available, the system would be enhanced with **Retrieval-Augmented Generation (RAG)** to ground the critique in specific internal documentation, making the "Devil's Advocate" persona more relevant.

To evaluate the quality of collaboration, I would collect:

1. **Friction-to-Resolution Ratio:** Measuring the number of conversational turns required before code is generated.
2. **Semantic Diversity Score:** Measuring the difference between solutions generated by different dyads to test for homogenization.
3. **Agency Self-Reports:** A post-task survey assessing whether the user felt they were the "Architect" or the "Operator."