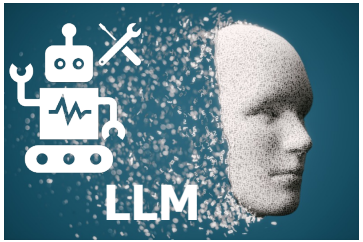
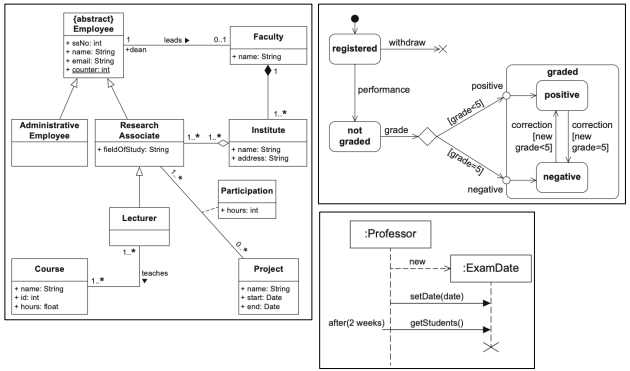
Intelligent Tutoring



+

UML Modeling



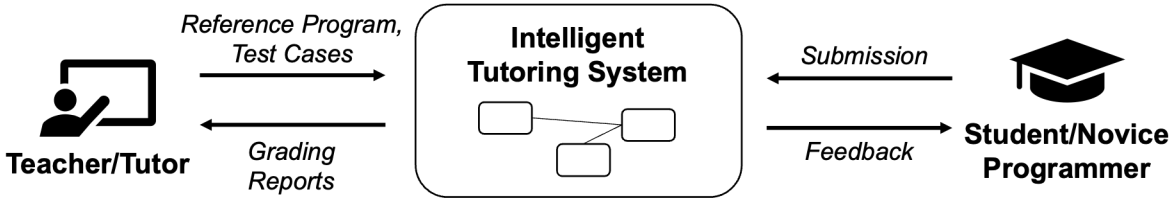
RUHR-UNIVERSITÄT BOCHUM

Intelligent Tutoring System

Exploring LLMs to develop an intelligent tutor for UML modeling

Samra Mehboob, M.Phil.
Prof. Dr. Yannic Noller
Software Quality group

Motivation and Background (1/2)

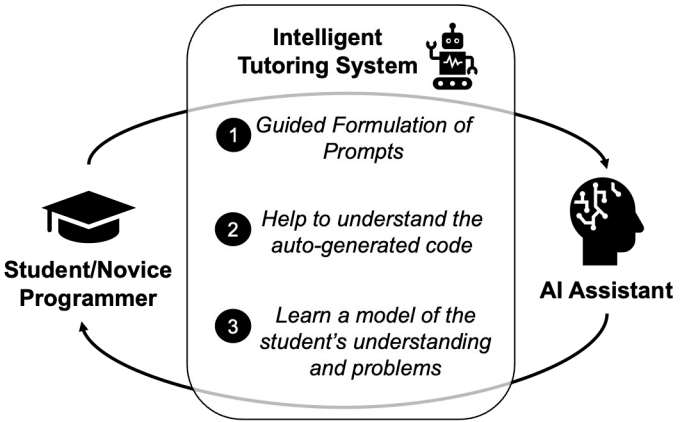


- The growing number of students enrolling in Computer Science (CS) programmes is pushing CS educators to their limits. This poses significant challenges to computing education, particularly the teaching of introductory programming and advanced software engineering (SE) courses.
- Intelligent Tutoring Systems (ITS)[1] provide automated, real-time feedback for novice students in programming courses to support the CS educators!

[1] Zhiyu Fan, Yannic Noller, Ashish Dandekar, and Abhik Roychoudhury. "Software Engineering Educational Experience in Building an Intelligent Tutoring System.“, To be presented at CSEE&T 2025. https://arxiv.org/pdf/2310.05472

Motivation and Background (2/2)

- With the shift from manual programming to AI-assisted programming, CS education must also be innovated.
- The ITS represents a well-suited platform to help students learn an effective way of using AI-based code generation tools like GitHub Copilot[2] and ChatGPT[3].
- Therefore, instead of exposing the student directly to the AI assistant, the ITS can *moderate* the prompts and explain the generated code, achieving a three-way interaction between the student, ITS, and AI assistant.

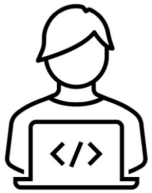


[2] https://github.com/features/copilot

[3] https://chatgpt.com/

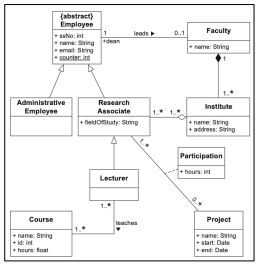
Project Goal

- So far, Intelligent Tutoring System (mostly exclusively) focused on helping with programming assignment. This project, will explore various ways of supporting other elements of the Software Engineering pipeline.
- This year, we start with UML modeling. The particular focus is to focus on providing feedback for modeling UML class diagrams and state charts.



Student

Please provide me feedback for this design!



Thanks! This already helps to realize feature X. But you might run into extensibility issues. Why don't you try to use the design pattern Y?

Assistant

Student Task and Responsibilities (1/2)

- This project is research-related but still implementation-heavy. As a start, in this project, the students can focus on UML class diagrams only.
- First, students need to formalize a given model so that the Large Language Model (LLM) can understand and reason about the model.
- Two recommended starting points:
  - the PlantUML[4] model language that is also supported by IntelliJ, or
  - the popular Java-based UML model editor UMLet[5]
- The students should explore at least one of these starting points. After getting familiar with the project, the students need to hook into the existing implementation (i.e., the standalone model editor or in the IDE extension) and add a feature to trigger the LLM.

[4] https://plantuml.com/

[5] https://github.com/umlet/umlet

Student Task and Responsibilities (2/2)

- Then the formalized model, i.e., transformed from the internal PlantUML/UMLet model into something the LLM can understand, will be send to the LLM together with appropriate prompts. Note: this is non-trivial!
- The students are expected to explore various prompting options and document them systematically.
- Additionally, the feedback must be also shown to the student.
- A straightforward solution could be some sort of textual comment. Even such simple textual comment must be vetted because the solution should never be directly exposed to a student. We only want to show hints and explanations to students, but no solutions!
- The implementation(s) must be evaluated on available SE modeling tasks, which can be selected by the students. The findings of the conducted experiments must be thoroughly documented.

Minimum Expectation and Extensions

- The minimum expected result is the implementation of an extension of PlantUML/UMLet that supports the described automated functionality to formalize a class diagram, prompt an LLM about feedback, and display this feedback to the student.
- This is a research-related project, so the students are expected to explore various design options and document them properly.
- An important aspect of this project is the design of the implementation, the maintainability, and the extensibility. Further, the implementation should be properly tested.
- The group must show the operationability of their implementation with a live demonstration.
- Potential extension points:
  - The group can illustrate the feedback more interactively, e.g., by adding “comments“ into the UML model, or by suggesting changes directly in the model that can be accepted/rejected by the user.
  - The group can explore the performance of multiple LLMs as an experimental evaluation.

Initial Timeplan

Week	Date	Phase	Milestones	Comment
1	13/04/2026	Kick-Off & Introduction	M1: Project Kick-Off	*(first week)*
2	20/04/2026	Planning, Requirements Engineering, and Design		*(last chance to de-register from studyproject)*
3	27/04/2026		M2: Task Breakdown and Code Design	
4	04/05/2026	**1st Coding Cycle** Implementation of prototype (no complete workflow implementation expected)		
5	11/05/2026			
6	18/05/2026		M3: Demonstration of Prototype	
7	25/05/2026	**2nd Coding Cycle** Implementation of complete workflow		*Whitsun/Pentecost Holidays*
8	01/06/2026			
9	08/06/2026		M4: Demonstration of Complete Workflow	
10	15/06/2026	**3rd Coding Cycle** Improvements and extensions		
11	22/06/2026			
12	29/06/2026		M5: Demonstration of Improved Workflow	
13	06/07/2026	**Finalization code** (code freeze after week 16)	M6: Final Code Submission	
14	13/07/2026	Final **documentation** and report writing/submission	M7: Report Draft Submission	
15	20/07/2026		M8: Final Report Submission	External deadline: TBA
			extra: Presentation in SQ Colloquium	

Working Mode

- Weekly meetings with advisor (will be arranged taking into account all schedules)
- Expected are at least one additional weekly group-internal meeting and active discussions on Slack/Discord
- Kick-Off Meeting: in the week of 13th April 2026 (details will be announced)

Other Information

- **Prerequisites:** Programming experience, preferably in Java, is needed for this project. Further, having experience with UML modeling (i.e., class diagrams and state charts) would be beneficial. Prior knowledge of using LLMs is a plus but not needed. Prior attendance of the Software Engineering course is highly recommended.
- **Deliverables:** Source code, its documentation, and a publishable report (incl. the evaluation results), ideally to submit to a conference (e.g., as a Demo paper) or at least publish as a technical report on arxiv.org.
- **Number of Participants:** 2-4
- **Target Group:** Master students
- **Industrial partner:** None (done at RUB)

Contact

Samra Mehboob, M.Phil.

Research Associate / PhD Student

- Room: MC 4.115
- E-Mail: samra.mehboob@rub.de
- Office hours: By Arrangement

- http://www.linkedin.com/in/samramehboob6816b7135/