Week 2: Paper 2: Towards AI-Native Software Engineering (SE 3.0): A Vision and a Challenge Roadmap

Haonan Zhang
h294zhang@uwaterloo.ca

1 Problem being solved

Software Engineering 2.0 (SE 2.0), which incorporates AI copilots into traditional development workflows, has several significant limitations. Developers often suffer a heavy cognitive burden, as they still lead the coding process. While AI copilots can automate certain tasks, they frequently generate buggy or inefficient code, creating additional maintenance challenges. The reliance on data-driven models further compounds the issue, as these models are resource-intensive to train and lack a deep understanding of domain-specific tasks. Based on the current practice in SE 2.0, the authors highlight the key challenges we need to address to shift to a new era of Software Engineering (SE 3.0).

2 New idea

The authors introduce what it looks like in SE 3.0 and the challenges we need to address to reach this new era of software engineering. SE 3.0 shifts away from task-driven approaches to an intent-first, conversation-oriented development process. In this vision, AI copilots evolve into fully collaborative AI teammates, equipped with advanced reasoning capabilities and designed to work symbiotically with human developers. To support this transformation, the authors outline the SE 3.0 technology stack, which includes key components: Teammate.next, a system of personalized and adaptive AI collaborators; IDE.next, an intent-driven integrated development environment; Compiler.next, which enables multi-objective code synthesis and optimization; Runtime.next, designed for SLA-aware execution that leverages edge computing; and FM.next, a curriculum-driven model training approach aimed at improving efficiency and reasoning capabilities.

However, achieving SE 3.0 comes with several challenges. One important issue is finding the right balance in how to ask clear questions. Asking too much can disrupt the workflow, while too few may lead to misunderstandings or incomplete guidance during interactions. Another challenge is improving the efficiency of code synthesis processes to deliver faster and more accurate results without sacrificing quality. Additionally, enhancing runtime performance is important, particularly in designing systems that can respond to our requirements timely. Addressing these issues will require creative solutions and a fundamental shift in how AI is integrated into software engineering practices.

3 Positive points

1. The authors provided a clear and detailed definition of our current practices in Software Engineering, termed SE 2.0, which integrates AI-assisted tools like copilots into traditional SE workflows. They highlighted the strengths and limitations of SE 2.0, creating a strong foundation for understanding the need for evolution.
2. The paper introduced a compelling vision for Software Engineering in the future, termed SE 3.0. They also introduced the technical stack required and challenges that need to be addressed to reach SE 3.0, which can inspire researches that facilitate the process of moving on.

4 Negative points

1. While the paper presents a visionary framework for SE 3.0, it lacks concrete implementation details. Many of the proposed components, such as IDE.next and Compiler.next, are described in abstract terms without providing actionable steps or prototypes, making it challenging to understand what they really look like.
2. The paper relies heavily on hypothetical scenarios and idealized outcomes to describe the benefits of SE 3.0. This makes it difficult to assess the practicality of the solutions in real-world, diverse software engineering environments.

5 Future work

Based on the negative points, in the future maybe they can elaborate more about some of their theories with some hands-on experiences or some examples.

6 Rating

4.5/5. This paper overall is good and insightful. 0.5 points are deduced because of the lack of practical experiences and examples.

7 Discussion points

1. Do you think SE 3.0 is just SE based on agents empowered by LLMs?
2. Considering there is already some research about developing an LLM-based agent to solve SE problems, do you agree that we are actually at SE 2.5?

3. Do you think the FMs are actually thinking or just doing content retrieval? If they are not actually thinking then can we ever reach SE 3.0?
4. What can SE researchers do in the era of LLM?

8 Class discussion

8.1 Top Universities vs. Industry

In the era of LLM, top universities often struggle to implement cutting-edge advancements, whereas industries can, due to their vast resources.

8.2 Are FMs thinking or retrieval?

A student disagrees with the notion that LLM thinks or creates from scratch. They argue, as the professor mentioned, that LLMs statistically output words (or code) based on user preferences rather than genuine reasoning.

8.3 Risks in the software industry.

The real risk in the software industry isn't AI itself but the human decisions behind its use. People often mistake AI challenges for "simple math problems", which leads to poor decisions. And when mistakes occur, it's not AI being "stupid" but rather humans mismanaging it.

8.4 Where are we now?

8.4.1 Presenter's view. The presenter thinks we are SE 2.5 since there are already many related kinds of research on using LLM-based agents to address the challenges.

8.4.2 First audience's view. We are at SE 1.9. Not everyone is using AI extensively, and it's not yet integrated across the entire industry.

8.4.3 Second audience's view. We are at SE 2.0. AI is helpful but has significant limitations. LLMs lack deep understanding and long-term memory. They don't debug code but instead suggest the most popular solutions, functioning like a StackOverflow rather than providing genuine debugging assistance.

8.4.4 Third audience's view. For graphics and video animation, we are at SE 2.5. She also mentioned that AI struggles with creative tasks like game development. For example, how would AI know where to start when creating a children's game?

8.5 What would education and the role of developers look like in a SE 3.0 world?

The professor notes that undergraduate curricula evolve slowly compared to industry needs. A student suggests that FMs are still important for education and grading systems might evolve to evaluate students based on prompts or conversational skills with AI.

8.6 How would human teammates collaborate with AI in SE 3.0?

Could there be a "Super AI teammate" where all software engineers interact with one advanced AI? This AI would serve as a conversational partner to discuss and refine project intents, acting as a central hub for collaboration.

Acknowledgments

Big thanks to Paul Wooseok Lee for the awesome class notes. Shoutout to Professor Godfrey and everyone in the class for the great discussions and thoughtful insights!