Rebuilding AI from the Small: The Path Toward Democratized Intelligence

Antonio Della Porta*
University of Salerno
Salerno, Italy
adellaporta@unisa.it

Abstract

AI is reshaping Software Engineering, but the tools with the highest capabilities remain difficult for many researchers and organizations to rely on. Some are accessible only through closed interfaces that change unpredictably, hindering reproducibility and long-term scientific progress. Others require hardware far beyond the reach of typical academic labs or industry teams, limiting who can meaningfully participate in this technological shift. Small Language Models offer a practical middle ground: they are open, efficient, and deployable in controlled environments. Yet they continue to lag behind larger models in tasks requiring deep reasoning. This dissertation aims to bridge that gap by studying how the form and structure of prompts influence model performance, and by developing new reasoning frameworks inspired by principles of human cognition. By enhancing the problem-solving abilities of smaller models, this work supports a future in which advanced AI capabilities are accessible, auditable, and usable by the broader SE community.

CCS Concepts

• **Software and its engineering** → **Software development techniques**; • **Computing methodologies** → **Artificial intelligence**.

Keywords

Small Language Models, Software Engineering, Cognitive Theories

ACM Reference Format:
Antonio Della Porta. 2026. Rebuilding AI from the Small: The Path Toward Democratized Intelligence. In *2026 IEEE/ACM 48th International Conference on Software Engineering (ICSE-Companion '26), April 12–18, 2026, Rio de Janeiro, Brazil.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3774748.3787626

1 Introduction

The Generative AI revolution began as a shared scientific endeavor, a promise of open tools to augment human intellect. Today, that promise is broken and the field is now dominated by massive, proprietary, closed-source models (e.g., ChatGPT, Claude, Gemini). They showed the potential to augment or automate tasks across the entire development lifecycle, but these promises are only available to the ones that are able to afford those.

This trend, which I term the *"SaaS-ification of intelligence"*, creates challenges for the entire Software Engineering community. First of all, it makes our empirical research potentially unverifiable, as we cannot build reliable science on black-box APIs that are updated at whim, invalidating our results overnight. Furthermore, many organizations are bound by strict contractual and regulatory obligations (e.g., GDPR, HIPAA) governing how data is managed, processed, and transmitted. Sending an entire proprietary codebase to a third-party API (even a private tier) may breach these contracts and create an unacceptable security and privacy liability.

In response to this crisis, researchers have begun to return to the origins of the GenAI endeavor, providing open-weight models to everyone to use. This movement, driven by a philosophy of open science, is now achieving increasingly higher performance in SE tasks, with top-tier open models rivaling the best closed-source competitors. This hard-won victory has revealed a new, practical, and equally prohibitive hurdle: the "Size Barrier".

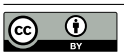
An "open" 1T-parameter model[6] is a mirage of democratization; it is "open" in name only since it remains profoundly inaccessible to the vast majority of university labs or startups. The capital expenditure for the massive, multi-million dollar GPU clusters required to simply run inference—let alone fine-tune—these models creates a new *"intelligence divide"* based on access to hardware.

This research posits that the *only* realistic and sustainable path to true democratization lies with Small Language Models (SLMs). While the literature lacks a unified consensus on a precise threshold, I follow recent work in defining SLMs[5, 9] as models (often less than 15-20B parameters, e.g., Phi-3, Llama 3 8B) explicitly designed for high performance and accessibility on commodity hardware. SLMs are now one of the most promising solution to both solve the SaaS-ification crisis (they are open and private) and the size barrier (they are efficient and accessible).

Not all that glitter is gold however, since the SLMs have a capability gap when compared to traditional LLMs in complex or context-heavy context, that represents skills needed in real-world SE tasks like repository-level bug fixing or architectural design. This capability gap is easily noticeable looking at the score obtained in reasoning-intensive benchmarks like SWE-bench, which require models to resolve real-world GitHub issues.

This reasoning gap itself provides a clue. Recent studies by Dasgupta et al.[3] and Lampinen et al.[7] demonstrate that LLMs, like humans, display content-dependent reasoning biases. Researchers from Anthropic[1] have also analyzed the model's ability to internally represent its own knowledge and uncertainty, discovering that they indeed shows signs of these capabilities. These articles integrate prior knowledge and beliefs, blurring the line between

*2 out of 3 years PhD Student, advised by Fabio Palomba and Stefano Lambiase



ICSE-Companion '26, Rio de Janeiro, Brazil

ACM ISBN 979-8-4007-2296-7/2026/04
https://doi.org/10.1145/3774748.3787626

symbolic logic and experiential understanding. This parallel suggests that, since LMs reason more like humans than like simple compilers, the most effective way to close their performance gap may not be through a single perfect instruction, but through the same human-like processes of refinement, collaboration, and structured thought that define human intelligence.

Goal of the Ph.D Project

Help the process of democratization of the capabilities of LMs by designing robust frameworks that unlock the problem-solving power of small models, making a truly private, accessible, and democratic AI a practical reality for both researchers and practitioners in software engineering.

2 Research Objective

To address the goal proposed, this Ph.D. project research follows two distinct objectives:

RO_1: Analyzing the Input-Centric Path: Limits and Opportunities of Prompt Optimization

How far can prompt engineering improve the performances of SLMs and what are their limits?

This research objective focuses on understanding how the design and structure of prompts influence the performance and output quality of SLMs on software engineering tasks. The first component of this investigation evaluates the extent to which prompt-based interventions, such as Few-Shot [2], Chain-of-Thought [10], ReAct [12], and Self-Refine [8], affect the model's ability to generate coherent and functionally meaningful outputs. The intention is not to assume these techniques are inherently beneficial, but to measure their actual impact across different categories of tasks and to determine the points at which they stop yielding improvements.

The second component analyzes the intrinsic properties of prompts. This includes examining syntactic characteristics (e.g., sentence structure, prompt length, presence of code or formal notation) and semantic dimensions (e.g., level of abstraction, clarity, specificity). By systematically studying these attributes, the objective is to identify which prompt features are most useful for the creation of more useful and higher-quality outputs in SLMs.

RO_2: Building the Process-Centric Path: The Opportunities of Cognitive Theories on LLMs

How effective are cognitive theories in improving performances of SMLs?

This objective aims to understand, propose, and validate the impact of cognitive theories on the reasoning performances of SLM, and to determine whether the existing performance gap can be reduced through new frameworks grounded in these theories. The goal is to measure how these human-like reasoning processes improve model performance and output quality on complex SE tasks, and to define the boundaries and true potential of this new, more robust approach.

This research objective investigates whether the reasoning capabilities of SLMs can be enhanced by incorporating structured reasoning processes inspired by cognitive theories. The emphasis here is not on modifying the prompt itself, but on shaping the reasoning workflow the model executes while solving a task. This includes operationalizing principles such as systematic problem decomposition, guided reflection, and self-questioning into explicit frameworks that structure how the model approaches inference.

The aim is to develop and evaluate new reasoning procedures grounded in these cognitive principles and to test whether they help SLMs overcome limitations that prompt engineering alone cannot address. The assessment focuses on complex software-engineering tasks where robust reasoning is essential, measuring how process-centric interventions affect accuracy, consistency, and the interpretability of generated outputs.

3 Expected Contribution and Evaluation Plan

The expected contributions of this project, driven by the research objectives described, will follow three different but complementary paths:

Prompt Composition Analysis. The first contribution will be a systematic and experimentally grounded understanding of how different parts of a prompt influence the performance and reasoning of SLMs. To achieve this, the project will build a large dataset of controlled prompt variations, where prompts are decomposed into their syntactic and semantic components. Each component will be measured for its effect on output quality across a variety of SE tasks. This will allow us to identify which linguistic structures, levels of specificity, forms of context, and kinds of reasoning cues actually lead to better or worse results.

Verification and Validation of Prompts. A second, tightly connected contribution will be the creation of a verification and validation process for prompts, something the SE community currently lacks. This process will include automated checks, structural analyses, and quality metrics that allow researchers and practitioners to evaluate a prompt before using it in real-world workflows. The outcome will be a practical method for testing whether a prompt is clear, stable, and robust enough to produce reliable outputs. This contribution is intended to give prompt engineering a more disciplined and reproducible foundation and to support the responsible use of SLMs in research and industry.

Cognitive Reasoning Frameworks for SLMs. The third contribution will be the design and implementation of new reasoning frameworks for SLMs that are inspired by cognitive theories. These frameworks will be built as concrete procedures that guide a model through structured reasoning, such as breaking down a task into smaller steps, performing intermediate checks, or reflecting on previous attempts. Each framework will be implemented as an inference-time pipeline that can be applied to any SLM without retraining. Their effectiveness will then be validated on complex SE tasks like bug fixing, repository-level reasoning, and multi-step code generation.

4 Preliminary Results

During the first year of the Ph.D. project, my work has focused on RO_1 and the problem of prompt optimization. The main accepted contribution is an article presented at the *29th International Conference on Evaluation and Assessment in Software Engineering (EASE '25)*, in which we investigate how different prompt patterns affect the quality of generated code. The study examines whether structured prompt templates can improve key attributes such as maintainability, security, and reliability[4]. Using the Dev-GPT[11] dataset, we analyzed a large corpus of real prompts paired with model-generated code, identified the most widely used prompting strategies, and assessed their impact through standard software quality metrics. The results show that, although patterns such as Zero-Shot, Chain-of-Thought, and Few-Shot are common in practice, they do not lead to measurable differences in the assessed quality attributes. This finding suggests that, at least in this context, prompt structure alone has a limited effect on the quality of generated code.

Building on this line of research, I have extended the investigation in two additional articles currently under submission.[1]

The first one investigates which characteristics of developer-written prompts actually influence the quality of code produced by Large Language Models. Using real single-turn prompts collected from GitHub, the study analyzes prompts in terms of their readability and structural properties and examines how these features relate to the usefulness, correctness, and semantic similarity of the generated code. Through regression analysis, the work shows that readability strongly predicts usefulness and correctness, while structural aspects of the prompt influence similarity to real committed code. The paper provides early evidence that prompt quality is multi-dimensional and that measurable prompt characteristics can help explain variations in code-generation outcomes.

The second article investigates how the internal syntactical components of prompts may affects the security of generated code. Starting from a dataset of security-related prompts, we generate systematic syntactic variations for each prompt and use them to produce code in multiple languages. The resulting programs are analyzed with CodeQL to measure the presence of security issues. The study shows that specific structural components of prompts, as well as their position within the prompt, reliably influence the likelihood of introducing vulnerabilities.

References

[1] Anthropic. Code execution with mcp: Building more efficient agents. https://www.anthropic.com/engineering/code-execution-with-mcp, 2025.

[2] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems 33* (2020), 1877–1901.

[3] Dasgupta, I., Lampinen, A. K., Chan, S. C., Creswell, A., Kumaran, D., McClelland, J. L., and Hill, F. Language models show human-like content effects on reasoning. *arXiv preprint arXiv:2207.07051 2*, 3 (2022).

[4] Della Porta, A., Lambiase, S., and Palomba, F. Do prompt patterns affect code quality? a first empirical assessment of chatgpt-generated code. *arXiv preprint arXiv:2504.13656* (2025).

[5] Hasan, M. M., Waseem, M., Kemell, K.-K., Rasku, J., Ala-Rantala, J., and Abrahamsson, P. Assessing small language models for code generation: An empirical study with benchmarks. *arXiv preprint arXiv:2507.03160* (2025).

[6] InclusionAI. Ling-1t. https://huggingface.co/inclusionAI/Ling-1T, 2025.

[7] Lampinen, A. K., Dasgupta, I., Chan, S. C., Sheahan, H. R., Creswell, A., Kumaran, D., McClelland, J. L., and Hill, F. Language models, like humans, show content effects on reasoning tasks. *PNAS nexus 3*, 7 (2024), pgae233.

[8] Madaan, A., Tandon, N., Gupta, P., Hallinan, S., Gao, L., Wiegreffe, S., Alon, U., Dziri, N., Prabhumoye, S., Yang, Y., et al. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems 36* (2023), 46534–46594.

[9] Wang, F., Zhang, Z., Zhang, X., Wu, Z., Mo, T., Lu, Q., Wang, W., Li, R., Xu, J., Tang, X., et al. A comprehensive survey of small language models in the era of large language models: Techniques, enhancements, applications, collaboration with llms, and trustworthiness. *ACM Transactions on Intelligent Systems and Technology* (2024).

[10] Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q. V., Zhou, D., et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems 35* (2022), 24824–24837.

[11] Xiao, T., Treude, C., Hata, H., and Matsumoto, K. Devgpt: Studying developer-chatgpt conversations. In *Proceedings of the 21st international conference on mining software repositories* (2024), pp. 227–230.

[12] Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K. R., and Cao, Y. React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations* (2022).

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009

[1] Venues omitted for double blind reasons.