

CHAPTER 5 – CONCLUSIONS AND RECOMENDATIONS

5.1 Conclusions

This study set out to explore the integration of Large Language Models (LLMs), specifically ChatGPT, as co-creators in the software development process, using the case of the MBKM system at Swiss German University (SGU). The research aimed to evaluate how LLMs can assist in various phases of software development, and to compare the AI-assisted development process with traditional approaches. Based on the research questions and objectives, the following conclusions are drawn:

Research Question 1: How can Large Language Models (LLMs) be effectively utilized as co-creators in the development of the MBKM system at Swiss German University to enhance the software development process and meet the program's requirements?

ChatGPT was employed across the Software Development Life Cycle (SDLC), assisting in requirement elicitation from interview transcripts, system modelling (use case, activity, and class diagrams), code generation, and test case creation. Its strength lies in accelerating structured output generation, supporting developers in rapidly transforming abstract requirements into actionable software artifacts. When guided by clear prompts and domain context, ChatGPT served as a highly effective co-creator throughout the development of the MBKM system.

Research Question 2: What are the challenges and limitations associated with incorporating LLMs in the software development process for the MBKM system?

While ChatGPT excels at text-based outputs (requirements, code, test cases), it requires strong contextual grounding and clear objectives to produce useful results. It also struggles with generating visual content, such as accurately

organized UML diagrams or screen layouts, limiting its usefulness in visual design phases.

Research Question 3 How does the integration of LLMs impact the efficiency, specifically in terms of reducing development time, improving documentation completeness, and enhancing overall user acceptance of the system?

The integration of LLMs into the development of the MBKM system had a notable impact on efficiency and accuracy. Documentation completeness improved by 26.7%, primarily due to ChatGPT's ability to generate artifacts across the SDLC. Development time was reduced by 60%, with the most significant time savings observed in requirement engineering, documentation generation, and testing preparation. While ChatGPT significantly accelerated the development process, there was no substantial difference in the final overall user satisfaction.

This study concludes that LLM-assisted software development when guided effectively can enhance the delivery and quality. LLMs function best not as replacements, but as collaborative agents that complement human judgment and creativity.

5.2 Future Works

Several areas remain open for future exploration to deepen understanding, extend applicability, and improve best practices in AI-assisted software engineering.

Expansion to Other Domains and Systems.

This study was limited to the development of an academic portal (MBKM system). Future research can replicate this AI-assisted development approach in other domains to examine whether similar can be achieved across different types of software systems.

Security and Ethical Evaluation of LLM Outputs.

LLMs can produce insecure logic patterns, data leakage risks, or biased assumptions. Future studies should evaluate AI-generated outputs using formal security audits to ensure the safe and ethical use of AI in software engineering workflows.

Broader Evaluation Metrics.

While this study used usability testing, future research should explore additional performance and completeness indicators.

Exploration of Alternative and Specialized AI Models.

While ChatGPT served as the primary co-creator in this study, future research can explore the use of alternative LLMs and compare their performance in software development contexts. Each model may offer unique advantages in reasoning, creativity, or response time. In addition, domain-specific AI models tailored for tasks such as code generation, database design, or UI prototyping could be integrated into toolchains to support more specialized development workflows.

Autonomous and Collaborative AI Agents

Future research may also investigate the use of AI agents capable of performing multiple, interdependent tasks autonomously within a software engineering pipeline.

Scalability of AI-Assisted Development.

This research was conducted with a limited number of developers. Future work should evaluate how AI-assisted development scales when applied to larger teams and organizations. This includes studying how collaboration evolve when multiple developers interact with AI simultaneously. Such investigations would provide insights into how LLMs can support enterprise-scale development while maintaining efficiency and consistency.

By pursuing these directions, future researchers and practitioners can better understand how to scale, secure, and optimize LLM-assisted software development, contributing to more intelligent and adaptive engineering processes.

GLOSSARY

AI (Artificial Intelligence): The simulation of human intelligence processes by machines, especially computer systems, including learning, reasoning, and self-correction.

ChatGPT: A conversational AI developed by OpenAI based on the GPT (Generative Pre-trained Transformer) architecture, used in this study as a co-creator in the software development process.

FDD (Feature-Driven Development): An Agile software development methodology that organizes work around the delivery of client-valued features.

LLM (Large Language Model): A deep learning model trained on extensive text data, capable of generating and understanding human-like text, such as ChatGPT, Claude, LLaMA, and others.

MBKM (Merdeka Belajar-Kampus Merdeka): An Indonesian Ministry of Education policy that allows students to participate in experiential learning activities outside their primary study program for up to 60 credits.

Prompt Engineering: The technique of crafting clear and specific instructions (prompts) to elicit accurate and useful outputs from language models.

Prototype: An initial version of a system built to validate concepts and gather feedback for future improvements.

Requirement Elicitation: The process of identifying user needs and gathering requirements for a software system through various methods such as interviews and document analysis.

SDLC (Software Development Life Cycle): A structured process consisting of stages such as planning, analysis, design, implementation, testing, and maintenance, used to develop information systems.

SGU (Swiss German University): A private international university in Indonesia offering dual degree programs, which serves as the case study site for this thesis.

SRD (System Requirements Document): A formal document that outlines the functional and non-functional requirements of a system, serving as a blueprint for design and development.

System Artifact: Any tangible byproduct produced during the software development process, such as diagrams, code, wireframes, or documentation.

UI/UX (User Interface/User Experience): UI refers to the layout and interactive elements of a system, while UX focuses on the overall user experience, usability, and satisfaction.

Usability Testing: A method for evaluating how easy and effective a system is by testing it with representative users and collecting feedback.

REFERENCES

- Ahmad, A., Waseem, M., Liang, P., Fahmideh, M., Aktar, M.S. and Mikkonen, T. 2023. Towards Human-Bot Collaborative Software Architecting with ChatGPT. In: *Proceedings of the 27th International Conference on Evaluation and Assessment in Software Engineering*. New York, NY, USA: ACM, pp. 279–285. doi: 10.1145/3593434.3593468.
- Alsaqqa, S., Sawalha, S. and Abdel-Nabi, H. 2020. Agile Software Development: Methodologies and Trends. *International Journal of Interactive Mobile Technologies (iJIM)* 14(11), p. 246. doi: 10.3991/ijim.v14i11.13269.
- Arora, C., Grundy, J. and Abdelrazek, M. 2024. Advancing Requirements Engineering Through Generative AI: Assessing the Role of LLMs. In: *Generative AI for Effective Software Development*. Cham: Springer Nature Switzerland, pp. 129–148. doi: 10.1007/978-3-031-55642-5_6.
- Azaria, A., Azoulay, R. and Reches, S. 2023. ChatGPT is a Remarkable Tool -- For Experts.
- Bencheikh, L. and Höglund, N. 2023. *Exploring the Efficacy of ChatGPT in Generating Requirements: An Experimental Study*.
- Bsharat, S.M., Myrzakhan, A. and Shen, Z. 2023. Principled Instructions Are All You Need for Questioning LLaMA-1/2, GPT-3.5/4.
- Cámara, J., Troya, J., Burgueño, L. and Vallecillo, A. 2023. On the assessment of generative AI in modeling tasks: an experience report with ChatGPT and UML. *Software and Systems Modeling* 22(3), pp. 781–793. doi: 10.1007/s10270-023-01105-5.
- Chen, A. et al. 2023. Improving Code Generation by Training with Natural Language Feedback.
- Ekin, S. 2023. Prompt Engineering For ChatGPT: A Quick Guide To Techniques, Tips, And Best Practices. *TechRxiv*.
- Fan, A., Gokkaya, B., Harman, M., Lyubarskiy, M., Sengupta, S., Yoo, S. and Zhang, J.M. 2023. Large Language Models for Software Engineering: Survey and Open Problems.

-
- Hörnemalm, A. 2023. *ChatGPT as a Software Development Tool The Future of Development*. Umeå University.
- IEEE Computer Society. 2014. *Guide to the Software Engineering Body of Knowledge SWEBOOK ® A Project of the IEEE Computer Society*.
- Islam, R. and Moushi, O.M. 2024. *GPT-4o: The Cutting-Edge Advancement in Multimodal LLM*. Available at: <https://github.com/openai/tiktoken>.
- Kolthoff, K., Bartelt, C. and Ponzetto, S.P. 2023. Data-driven prototyping via natural-language-based GUI retrieval. *Automated Software Engineering* 30(1), p. 13. doi: 10.1007/s10515-023-00377-x.
- Li, J., Li, G., Li, Y. and Jin, Z. 2023. Structured Chain-of-Thought Prompting for Code Generation.
- Liu, C., Bao, X., Zhang, H., Zhang, N., Hu, H., Zhang, X. and Yan, M. 2023a. Improving ChatGPT Prompt for Code Generation.
- Liu, Y., Le-Cong, T., Widyasari, R., Tantithamthavorn, C., Li, L., Le, X.-B.D. and Lo, D. 2023b. Refining ChatGPT-Generated Code: Characterizing and Mitigating Code Quality Issues.
- Liu, Z., Tang, Y., Luo, X., Zhou, Y. and Zhang, L.F. 2023c. No Need to Lift a Finger Anymore? Assessing the Quality of Code Generation by ChatGPT.
- Ma, Q., Wu, T. and Koedinger, K. 2023. Is AI the better programming partner? Human-Human Pair Programming vs. Human-AI pAIr Programming.
- Mandal, S. et al. 2023. Large Language Models Based Automatic Synthesis of Software Specifications.
- Marques, N., Silva, R.R. and Bernardino, J. 2024. Using ChatGPT in Software Requirements Engineering: A Comprehensive Review. *Future Internet* 16(6), p. 180. doi: 10.3390/fi16060180.
- Monteiro, M., Branco, B.C., Silvestre, S., Avelino, G. and Valente, M.T. 2023. End-to-End Software Construction using ChatGPT: An Experience Report.
- Myrzakhan, A., Bsharat, S.M. and Shen, Z. 2024. Open-LLM-Leaderboard: From Multi-choice to Open-style Questions for LLMs Evaluation, Benchmark, and Arena.
- Noever, D. 2023. Can Large Language Models Find And Fix Vulnerable Software?
-

Olausson, T.X., Inala, J.P., Wang, C., Gao, J. and Solar-Lezama, A. 2023. Is Self-Repair a Silver Bullet for Code Generation?

OpenAI. 2024. *Hello GPT-4o*. Available at: <https://openai.com/index/hello-gpt-4o/> [Accessed: 29 November 2024].

Rajbhoj, A., Somase, A., Kulkarni, P. and Kulkarni, V. 2024. Accelerating Software Development Using Generative AI: ChatGPT Case Study. In: *Proceedings of the 17th Innovations in Software Engineering Conference*. New York, NY, USA: ACM, pp. 1–11. doi: 10.1145/3641399.3641403.

Ronanki, K., Berger, C. and Horkoff, J. 2023a. Investigating ChatGPT's Potential to Assist in Requirements Elicitation Processes. In: *2023 49th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. IEEE, pp. 354–361. doi: 10.1109/SEAA60479.2023.00061.

Ronanki, K., Cabrero-Daniel, B. and Berger, C. 2023b. ChatGPT as a tool for User Story Quality Evaluation: Trustworthy Out of the Box?

Sakib, F.A., Khan, S.H. and Karim, A.H.M.R. 2023. Extending the Frontier of ChatGPT: Code Generation and Debugging.

Schäfer, M., Nadi, S., Eghbali, A. and Tip, F. 2023. An Empirical Evaluation of Using Large Language Models for Automated Unit Test Generation.

Siddiq, M.L., Santos, J.C.S., Tanvir, R.H., Ulfat, N., Rifat, F. Al and Lopes, V.C. 2023. Using Large Language Models to Generate JUnit Tests: An Empirical Study. doi: 10.1145/3661167.3661216.

Sun, W. et al. 2023. Automatic Code Summarization via ChatGPT: How Far Are We?

Tian, H., Lu, W., Li, T.O., Tang, X., Cheung, S.-C., Klein, J. and Bissyandé, T.F. 2023a. Is ChatGPT the Ultimate Programming Assistant -- How far is it?

Tian, Z., Chen, J. and Zhang, X. 2023b. Test-Case-Driven Programming Understanding in Large Language Models for Better Code Generation.

White, J. et al. 2023. A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT.

White, J., Hays, S., Fu, Q., Spencer-Smith, J. and Schmidt, D.C. 2024. ChatGPT Prompt Patterns for Improving Code Quality, Refactoring, Requirements Elicitation, and Software Design. In: *Generative AI for Effective Software Development*. Cham: Springer Nature Switzerland, pp. 71–108. doi: 10.1007/978-3-031-55642-5_4.

-
- Wu, T., He, S., Liu, J., Sun, S., Liu, K., Han, Q.-L. and Tang, Y. 2023. A Brief Overview of ChatGPT: The History, Status Quo and Potential Future Development. *IEEE/CAA Journal of Automatica Sinica* 10(5), pp. 1122–1136. doi: 10.1109/JAS.2023.123618.
- Xie, D., Yoo, B., Jiang, N., Kim, M., Tan, L., Zhang, X. and Lee, J.S. 2023. Impact of Large Language Models on Generating Software Specifications.
- Yuan, Z., Lou, Y., Liu, M., Ding, S., Wang, K., Chen, Y. and Peng, X. 2023. No More Manual Tests? Evaluating and Improving ChatGPT for Unit Test Generation.
- Zhang, S., Wang, J., Dong, G., Sun, J., Zhang, Y. and Pu, G. 2024. Experimenting a New Programming Practice with LLMs.



CURRICULUM VITAE

Dennis Reynaldo

Tangerang, Indonesia | dennis.reynaldo@student.sgu.ac.id | +62 812 1051 0155

Profile

Creative and analytical professional with over 8 years of experience in professional motion design, currently pursuing a Master's in Information Technology, with academic research focused on integrating Large Language Models (LLMs) into the Software Development Life Cycle. Passionate about solving problems through creative thinking and data-driven solutions.

Education

Swiss German University — Master of Information Technology

Expected Graduation: 2025

Universitas Multimedia Nusantara — Bachelor of Design

Graduation: 2016

Professional Experience

Media Editor – Storyboard Communications, Canada (2025 – Present)

Associate Animation Director – Kabar Media, Indonesia (2023 – Present)

Art Director – Arkala Studio, Indonesia (2017 - 2023)