

AI Assistant to Improve Experimentation in Software Startups Using Large Language Model and Prompt Engineering

Mario E. S. Simaremare^{1,*}, Henry Edison¹

¹Blekinge Tekniska Högskola, 371 79 Karlskrona, Sweden

Abstract

Software startup is a unique type of company with unique characteristics. On the one hand, they must offer innovative products appealing to customers to generate revenue and survive, but on the other hand, they are limited in resources, time, and experience. During the new product development, it is important to experiment with their original ideas. However, doing a meaningful experiment requires resources and challenges. A study on failed software startups shows that, despite its importance, many software startups skipped or did not experiment with their ideas. The study identifies 25 inhibitors spread in five experimentation stages.

In the last few years, Large Language Models (LLMs) have become a popular technology. The advancement of LLM has made it adopted into many parts of the software development cycle. Studies show that LLM also has been used to generate new innovative product ideas and to manage innovation. However, there is no investigation into the possibility of utilizing the power of LLM to help software startups do experimentation. Interactions to an LLM are done through prompts. During the interaction or session, a user will send one or more prompts in a zero-, one-, or few-shots to an LLM agent. Unfortunately, learning and using prompts effectively requires time and resources, things that software startups are scarce with.

In this project, we aim to help improve the experimentation process and address the inhibitors by leveraging the power of LLMs. There are five initial research questions and studies planned in the project. In the first step, we will investigate current experimentation practices, challenges, inhibitors, and the strategies used to circumvent them. Secondly, we will investigate how AI has been used in today's experimentation. Then, we will investigate the set of measurements available to measure the success of an experiment. The next step is to investigate how to support experimentation using LLMs followed by a validation sequence. The first form of support is a prompt guidebook to help software startups use an LLM agent to help their experimentation. The second form is an LLM-based assistant tailored specifically to guide the experimentation process.

Keywords

Software startup, experimentation, large language model, prompt engineering, startup

ICSOB '23: 14th International Conference on Software Business, November 27–29, 2023, Lahti, Finland

*Corresponding author.


✉ mario.simaremare@bth.se (M. E. S. Simaremare); henry.edison@bth.se (H. Edison)

🌐 <https://github.com/simaremare> (M. E. S. Simaremare); <https://www.henryedison.com/> (H. Edison)

🆔 0000-0002-7873-6363 (M. E. S. Simaremare); 0000-0002-9494-8059 (H. Edison)



© 2023 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

1. Problem Definition

Software startups are emerging companies in the software domain that aim to present innovative software solutions [1]. Software startups often face significant uncertainties in a highly competitive environment with limited resources, experience, and tight time constraints [2, 3]. Software startups develop and introduce innovative software solutions to enhance existing markets or establish new ones.

Studies show that experimentation is one fundamental approach to finding a successful and sustainable business idea [4, 5, 6]. Experimentation is an iterative build-learn-measure cycle of validating an idea and shaping it into a more mature idea. This process is crucial for software startups since they are limited in resources to execute the idea in a full-fledged manner. Experimentation could also help the software startups decide whether to continue the venture or abandon the idea to minimize loss. For example, a software startup can create a minimum viable product, make it available to the users, and ask for feedback [5]. The feedback may lead to a slight idea adjustment to total rejection. Based on the feedback, the software startup can decide whether to continue or abandon the idea. In the software engineering context, the concept of experimentation itself is aligned with the agile method, which is widely adopted by software startups [7].

Unfortunately, the adoption of experimentation in software startups is low. A study by Melegati et al. [8] explores and models the inhibitors of why software startups failed in adopting experimentation. The study identifies 25 inhibitors in the experimentation process. They are categorized into (1) inhibitors related to awareness of experimentation, (2) inhibitors related to experiment intention, (3) inhibitors related to valid experiment, (4) inhibitors related to valid analysis, and (5) inhibitors related to considering the experiment result. The number of cases in the second category is the highest, meaning most of the software startups had no desire to experiment with their original ideas in the first place. Out of the 25 identified inhibitors, the top inhibitors that happen very often are (1) over-focusing on the product and its perfection, (2) overconfidence in the idea, and (3) some initial shreds of evidence may make the founders believe that their ideas are valid; hence no further experiment is needed. These inhibitors are likely to have a latent existence in software startups.

Recently, an AI-assisted approach to developing new product ideas has emerged. The combination of humans and AI generates new innovative ideas. Large Language Models (LLMs), a subset of AI technology, have revolutionized natural language processing tasks by efficiently learning linguistic patterns from vast text data. Popular LLMs, such as OpenAI's GPTs¹, Google's BERT², Gemini³, and Meta's Llama⁴ employ transformer architectures to generate or understand human-like text across various applications, from chatbots to content generation. These models have remarkable capabilities in capturing context, nuance, and semantics. Their pre-training and fine-tuning methodology makes them versatile across diverse tasks without task-specific model modifications [9, 10].

In this project, we aim to improve the experimentation process and address the inhibitors by

¹<https://openai.com/research>

²<https://blog.research.google/2018/11/open-sourcing-bert-state-of-art-pre.html>

³<https://deepmind.google/technologies/gemini/introduction>

⁴<https://research.facebook.com/publications/llama-open-and-efficient-foundation-language-models/>

leveraging the power of LLMs. There are five initial research questions and studies. We plan to run the project using the design science approach [11].

2. Knowledge Gap

Experimentation in software startups is critical to validate ideas and create a sustainable business. A study based on the postmortem artifacts shows that many software startups failed to experiment with their ideas, which, to some extent, might lead them to a disastrous state [8]. Most software startups acknowledged the concept of experimentation but chose not to do it. The study identified 25 inhibitors to experimentation spread in various experimentation stages. Some are organizational inhibitors, some are internal, and others are environmental-related.

However, the study does not elaborate on how or what strategies the software startups tried to overcome the inhibitors. This knowledge is important as a lesson learned for future cases. Unfortunately, extending the study could be challenging since the main sources are postmortem artifacts. A similar study on the surviving software startups could fill the gap. The result would give valuable knowledge on how to address the inhibitors. This draws the first research question (RQ1), *"What are the practices, challenges, inhibitors, and solutions in experimentation in software startups?"*

In addition, further exploration of adopting AI technologies in experimentation could also be performed to learn how AI technology has been used and how useful they are. This knowledge will allow further investigation into what areas or aspects can be improved. This raises the second research question (RQ2), *"How do software startups incorporate AI technologies, especially LLMs, in new product development?"*

It is also important to have a clear guideline, framework, or set of measurements to quantify whether an experiment produces a meaningful result regardless of the consequences, continuation, or abandonment of the idea. This leads to the third research question (RQ3), *"How to measure whether an experiment produces a meaningful result or not?"*

LLMs have gained massive attention in the last few years due to their advancement and potential. The vast amount of training datasets makes LLMs very powerful and knowledgeable tools. LLMs have been integrated into many aspects of software engineering, from requirement elicitation to testing and debugging [12, 13].

The advancement of LLMs has been used in many stages in the software development cycle [13, 14], from requirement refinement to bug fixing. In the software startup context, LLMs have also been used to generate new product ideas [15] and manage innovation [16]. However, little is known about studies investigating how to leverage the power of LLMs to help software startups do experimentation. Reflecting on the capability of LLMs opens an opportunity to leverage it to guide software startups to experiment with their ideas and simultaneously avoid the inhibitors.

Interactions to an LLM are done through prompts. During the interaction or session, a user will send one or more prompts in a zero-, one-, or few-shots to an LLM agent [10, 17]. Shot refers to the number of inputs or contexts to get a desired output. LLMs are working with a probabilistic approach. This means two exact prompts may return different outputs in two separate sessions [18].

The zero-shot is a prompt without any prior context or example. This means the LLM agent will try to understand the context only based on the sole prompt. In most cases, the sole prompt lacks clear context. Hence, it is very likely the zero-shot prompt will return a generic output that is less desirable for specific needs or cases.

The one-shot prompt is when the user communicates exactly one example or a context before sending the intended prompt. Compared to the zero-shot, this prompt will produce outputs closer to the expectation, a more desirable.

The few-shots prompt is when the user has supplied multiple examples or contexts before the intended prompt. This will very likely produce a more relevant output compared to the one-shot. This prompt is also called many-shots when the number of examples or context is very large.

An effective prompt strategy helps the LLMs produce the most relevant output [19]. However, writing prompts to leverage an LLM agent to guide the experiment process requires time and resources, things that software startups do not have. This raises another research question (RQ4), *"How to effectively use prompts to guide the experimentation process?"*

Moreover, the availability of previous valuable experiences from failed and survived software startups may improve the LLMs' responses. However, it has to be admitted that LLMs can only give directions or suggestions and not direct solutions based on a probabilistic approach. The final decision lies in the hands of the software startup's leaders. These experiences can make a better experimentation context when supplied into LLMs. In addition to that, specific information related to a project could even make a more personalized environment for better-guided experimentation. These lead to our latest research question (RQ5), *"How to improve the experimentation process by leveraging the power of LLMs?"*

3. Research Method

To answer RQ1 and RQ2, we plan to conduct two parallel studies. The first one (S1) is an interview study with software startups to understand the current practices, challenges, inhibitors, and solutions for doing experiments. Furthermore, we would like to know how they manage the effect of inhibitors. The result will then be analyzed to see patterns or strategies that can be useful in future cases. We will also investigate how these software startups incorporate AI technologies in their experimentation, how satisfied they are, and what could be improved. A literature study (S2) will be executed to enrich the knowledge from previous research. We aim to publish at least two articles from the studies.

For RQ3, we will run another literature study (S3) to see the suggested metrics or practices to measure meaningful experiments. The result is expected to give us a mapping and better understanding of which metrics or practices should be used at a particular stage in an experiment. We aim to publish an article from the study.

These three studies should draw a better understanding of the challenges, inhibitors, and solutions in experimentation from a more holistic angle and how to measure the meaningfulness of an experiment. Based on this, we could identify which areas LLMs can be leveraged to make a meaningful impact. An immediate output of this is a prompt guidebook containing strategies and prompt templates (S4a). The guidebook will be tailored based on the knowledge found in

the earlier studies. Later, a formal validation, in the form of case studies, will be conducted with selected software startups (S4b). We aim to publish at least one article from the study.

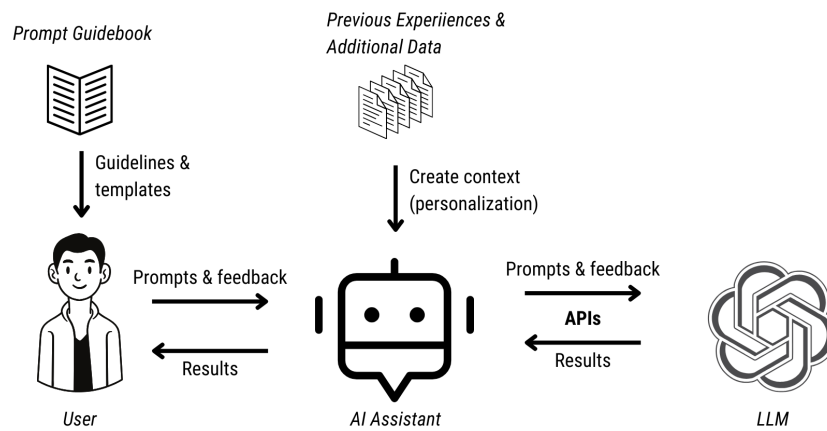


Figure 1: AI assistant, the general concept.

To address RQ5, we will further analyze the results of the earlier studies to identify which area of experimentation LLMs can support. Once decided, we will develop an LLM-based tool on top of one of the existing models. The tool will have the additional experiences from the earlier studies to generate relevant results (S5a). The tool can be further customized for specific needs through the few-shots approach. The users can also provide feedback in the form of relevancy level. The general concept of the tool can be seen in Figure 1. A fraction of the prompt guidebook may also be incorporated into the tool. Hence, these guides will be run automatically based on the prompt. Finally, a thorough evaluation and validation process will be together with selected software startups (S5b). The evaluation will be measured based on the users' satisfaction level (identified in the first study). We expect to publish at least two articles from the study.

In this research project, initially, we will not set clear boundaries or criteria for the software startups we want to work with. This makes us open to any opportunity that might show up during the research. In the future, boundaries might be introduced when necessary.

The whole project activity in the design science research is shown in Figure 2. The first three studies aim to develop a holistic understanding of the problem, in this case, the experimentation in the context of software startups and the use of AI, especially LLMs, in the experimentation process. The last two studies have a different aim, which is to develop possible solutions and validate them internally. Lastly, the solution will be brought in the real context but in a limited scope.

4. Timeline

This research project will run for five years. The detailed plan is shown in Table 1. The project started in early November 2023 and will run until November 2028. In the first year, we plan to run two studies, S1 and S2, to address RQ1 and RQ2. The studies are expected to be run

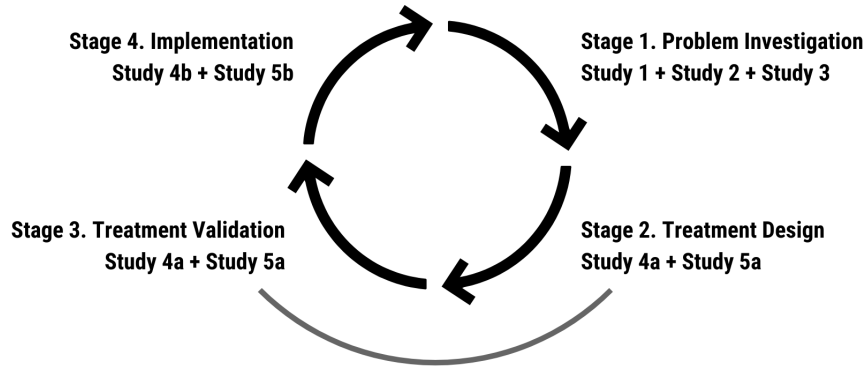


Figure 2: Project activities in the design science research framework.

in parallel. In the second year, we will do S3 to address RQ3. These three first studies are fundamental for the next two studies. Then, we will take the third year to run S4 to address RQ4. The remainder of the time will be used to run the final study, S5, and to address the last research question, RQ5.

Table 1
Project timeline from 2023 to 2028.

Research question	Activity	Period
RQ1: What are the practices, challenges, inhibitors, and solutions in experimentation in software startups?	S1: Interview study with software startups	2023-2024
RQ2: How do software startups incorporate AI technologies, especially LLMs, in new product development?	S1: Interview study with software startups, S2: Literature study	2023-2024
RQ3: How to measure whether an experiment produces a meaningful result or not?	S3: Literature study	2025
RQ4: How to effectively use prompts to guide the experimentation process?	S4a: Prompt guidebook dev. and S4b: Case study (for validation)	2025-2026
RQ5: How to improve the experimentation process by leveraging the power of LLMs?	S5a: AI assistant dev. and S5b: Case study (for validation)	2027-2028

5. Expected Contribution

There are at least five contributions from this research project, namely:

- A better knowledge of how software startups tackle challenges and inhibitors during experimentation. This knowledge will help future research on developing new strategies to reduce, if not eliminate, the known inhibitors.

- A better understanding of how far software startups have integrated AI technology into their experimentation pipeline. This will reveal which aspects of experimentation could be helped using the advancement of AI.
- A guideline or a framework or a set of measurements to help software startups judge the meaningfulness of an experiment.
- A validated prompt guidebook containing a guided walkthrough to help software startups do their experiments. With the help of additional contexts generated during the interaction, the walkthrough is expected to provide suggestions on which type of experiment the company should do and how to run the experiment.
- An AI assistant tool personalized to the software startup and a specific project where the company will experiment. The tool is expected to be validated together with selected software startups.

6. Acknowledgements

This work has been supported by ELLIIT; the Swedish Strategic Research Area in IT and Mobile Communications.

References

- [1] M. Unterkalmsteiner, P. Abrahamsson, X. Wang, A. Nguyen Duc, S. Shah, S. Sohaib, G. Baltes, K. Conboy, E. Cullina, D. Dennehy, H. Edison, C. Fernández, J. Garbajosa, T. Gorschek, E. Klotins, L. Hokkanen, F. Kon, M. I. Lunesu, M. Marchesi, A. Yagüe, Software Startups - A Research Agenda, *E-Informatica Software Engineering Journal* 2016 (2016) 89–124. doi:10.5277/e-Inf160105.
- [2] S. Sutton, The Role of Process in Software Start-up, *IEEE Software* 17 (2000) 33–39. URL: <https://ieeexplore.ieee.org/abstract/document/854066>. doi:10.1109/52.854066.
- [3] N. Paternoster, C. Giardino, M. Unterkalmsteiner, T. Gorschek, P. Abrahamsson, Software Development in Startup Companies: A Systematic Mapping Study, *Information and Software Technology* 56 (2014) 1200–1218. URL: <https://www.sciencedirect.com/science/article/pii/S0950584914000950>. doi:10.1016/j.infsof.2014.04.014.
- [4] S. H. Thomke, *Experimentation Matters: Unlocking the Potential of New Technologies for Innovation*, Harvard Business Press, 2003.
- [5] M. A. Hart, The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses Eric Ries. New York: Crown Business, 2011. 320 pages. US\$26.00., *Journal of Product Innovation Management* 29 (2012) 508–509. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1540-5885.2012.00920_2.x. doi:10.1111/j.1540-5885.2012.00920_2.x, [_eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1540-5885.2012.00920_2.x](https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1540-5885.2012.00920_2.x).
- [6] D. L. Frederiksen, A. Brem, How Do Entrepreneurs Think They Create Value? A Scientific Reflection of Eric Ries' Lean Startup Approach, *International Entrepreneurship and Management Journal* 13 (2017) 169–189. URL: <https://doi.org/10.1007/s11365-016-0411-x>. doi:10.1007/s11365-016-0411-x.

- [7] L. Williams, A. Cockburn, Agile Software Development: It's About Feedback and Change, *Computer* 36 (2003) 39–43. URL: <https://ieeexplore.ieee.org/document/1204373>. doi:10.1109/MC.2003.1204373, conference Name: Computer.
- [8] J. Melegati, H. Edison, X. Wang, XPro: A Model to Explain the Limited Adoption and Implementation of Experimentation in Software Startups, *IEEE Transactions on Software Engineering* 48 (2022) 1929–1946. URL: <https://ieeexplore.ieee.org/document/9282188>. doi:10.1109/TSE.2020.3042610, conference Name: IEEE Transactions on Software Engineering.
- [9] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2019. URL: <http://arxiv.org/abs/1810.04805>. doi:10.48550/arXiv.1810.04805, arXiv:1810.04805 [cs].
- [10] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, Y. Iwasawa, Large Language Models are Zero-Shot Reasoners, *Advances in Neural Information Processing Systems* 35 (2022) 22199–22213. URL: https://proceedings.neurips.cc/paper_files/paper/2022/hash/8bb0d291acd4acf06ef112099c16f326-Abstract-Conference.html.
- [11] R. J. Wieringa, *Design Science Methodology for Information Systems and Software Engineering*, Springer, Berlin, Heidelberg, 2014. URL: <https://link.springer.com/10.1007/978-3-662-43839-8>. doi:10.1007/978-3-662-43839-8.
- [12] I. Ozkaya, Application of Large Language Models to Software Engineering Tasks: Opportunities, Risks, and Implications, *IEEE Software* 40 (2023) 4–8. URL: <https://ieeexplore.ieee.org/document/10109345>. doi:10.1109/MS.2023.3248401, conference Name: IEEE Software.
- [13] C. Ebert, P. Louridas, Generative AI for Software Practitioners, *IEEE Software* 40 (2023) 30–38. URL: <https://ieeexplore.ieee.org/document/10176168>. doi:10.1109/MS.2023.3265877, conference Name: IEEE Software.
- [14] A. NguyenDuc, B. Cabrero-Daniel, C. Arora, A. Przybylek, D. Khanna, T. Herda, U. Rafiq, J. Melegati, E. Guerra, K.-K. Kemell, M. Saari, Z. Zhang, H. Le, T. Quan, P. Abrahamsson, *Generative Artificial Intelligence for Software Engineering - a Research Agenda*, 2023. URL: <https://papers.ssrn.com/abstract=4622517>. doi:10.2139/ssrn.4622517.
- [15] V. Bilgram, F. Laarmann, Accelerating Innovation With Generative AI: AI-Augmented Digital Prototyping and Innovation Methods, *IEEE Engineering Management Review* 51 (2023) 18–25. URL: <https://ieeexplore.ieee.org/document/10115412>. doi:10.1109/EMR.2023.3272799, conference Name: IEEE Engineering Management Review.
- [16] A. Brem, F. Giones, M. Werle, The AI Digital Revolution in Innovation: A Conceptual Framework of Artificial Intelligence Technologies for the Management of Innovation, *IEEE Transactions on Engineering Management* 70 (2023) 770–776. URL: <https://ieeexplore.ieee.org/abstract/document/9590499>. doi:10.1109/TEM.2021.3109983, conference Name: IEEE Transactions on Engineering Management.
- [17] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, D. Amodei, Language Models are Few-Shot Learners, 2020. URL: <http://arxiv.org/abs/2005.14165>. doi:10.48550/arXiv.2005.14165, arXiv:2005.14165 [cs].

- [18] L. Chen, M. Zaharia, J. Zou, How is ChatGPT's behavior changing over time?, 2023. URL: <https://arxiv.org/abs/2307.09009v2>.
- [19] L. Sun, Z. Shi, Prompt Learning Under the Large Language Model, in: 2023 International Seminar on Computer Science and Engineering Technology (SCSET), 2023, pp. 288–291. URL: <https://ieeexplore.ieee.org/document/10266544>. doi:10.1109/SCSET58950.2023.00070.