Beyond Code Assistance with GPT-4: Leveraging GitHub Copilot and ChatGPT for Peer Review in VSE Engineering

Roar Elias Georgsen[0000-0003-2868-497X]

University of South-Eastern Norway, Raveien 215, 3184 Borre, Norway
roar.e.georgsen@usn.no

Abstract. Most companies are Very Small Entities (VSEs), meaning they have fewer than 25 employees. Primarily domain specialists, these companies lack in-house expertise in important areas such as security and reliability engineering, process improvement, Quality Management (QM) and Systems Engineering (SE). VSEs struggle to adhere to Standard Operating procedures (SOP), and research has shown that contractual obligations to follow industry standards and best practices have little effect on actual engineering. This paper describes a case study that explored the potential of Large Language Models (LLMs) to support engineering best practices at a VSE by taking on the role of an expert peer in areas where the company had a skills gap. Aiwell, a Norwegian producer of building automation equipment, used ChatGPT, GitHub Copilot and GPT-4 to assess the quality of their system and stakeholder requirements. A GPT-4 foundation model with no additional training was given links to reference materials on requirements engineering produced by The International Council on Systems Engineering (INCOSE) and allowed to participate in discussions on the same digital collaboration platform as the human engineers. The study found that AI-assisted requirement reviews immediately and positively impacted the entire engineering process, supporting the feasibility of integrating advanced AI technologies in VSEs, even with limited training and resources. Participants highlighted the complementary nature of human intelligence and AI, where LLMs augmented human judgment through dialogue, leading to enriched engineering practices. Ethical and data privacy considerations also emerged as central themes, emphasising the need for proactive measures.

Keywords: generative AI· very small entities· systems engineering· requirements engineering

1 Introduction

The global supply chain consists mainly of Very Small Entities (VSE), with a workforce ranging from five (5) to twenty-five (25) people [1], and micro-enterprises, having fewer than nine employees, make up 92% of European enterprises [2]. In Norway, VSEs allocate approximately 30% of their spending to research and development (R&D), and new product development makes up 20-30% of VSE revenue, a rate significantly higher than the less than 9% observed in large companies [3, 4].

Aiwell, a small Norwegian company developing outdoor automation systems, wanted to modernise its engineering practices in response to the growing complexity of its customer projects. With a high workload and lack of senior engineers in the labour market, Aiwell wanted to explore how technology could mitigate potential skills gaps and improve quality. The seemingly intuitive ease with which individual engineers used tools like ChatGPT and GitHub Copilot to generate functioning code and documentation motivated Aiwell to initiate a pilot study.

1.1 Objectives of the Study

The study was designed to explore the potential of Large Language Models (LLMs) to enhance engineering practices in the context of Very Small Entities (VSEs). The specific objectives that guided this exploration were:

Impact on Productivity and Quality: Investigating how engineers without prior experience could utilise LLMs to increase productivity and improve quality in general.

Feasibility of Generative AI for VSEs: Assessing whether LLM-based tools could be integrated into a VSE workflow at an accessible cost and without additional human resources.

Impact on Engineering Competency: Investigating how LLMs could fill knowledge gaps and contribute to increased competency in VSEs, in particular with regard to requirements engineering.

By focusing on these key areas, the study intended to provide insights into how LLM-based tools can address specific challenges faced by VSEs.

2 Background

2.1 Engineering in Very Small Entities (VSE)

VSEs lack experience working with standardised processes [5] and often lean towards informal and organically evolved methodologies. They find standardised methods too broad for their specific needs and value the agility and the ability to tailor workflows that informality offers to suit their unique contexts [6]. Small companies are conscious of the escalating customer and legal expectations for systematic engineering and their internal need for improvement [5]. However, they do not view these as beneficial to their work or relevant to their situation [6]. This belief in the absence of added value leads to opposition to change, and the actual engineering practices may not align with documented compliance [7]. VSEs are rarely required to document complete compliance with specific standards and thus prefer to produce only the minimum documentation required by contractual obligations [6]. Measures such as risk-sharing partnerships [8] intended to improve system quality can, in this context, reduce the actual quality of the system as responsibility and accountability move down the supply chain.

Standardizing engineering practices can be vital to VSEs for a multitude of reasons. These practices equip VSEs with the resources and proven methodologies to enhance the quality and efficiency of engineering. They foster improved project management, reinforce implementation processes, and contribute to competitiveness [5]. The ISO/IEC 29110 series and similar initiatives, tailored for VSEs, can facilitate the transfer of codified knowledge related to systems engineering, offering benefits such as promoting innovation, market access, quality control, and ethical adherence.

Standards can provide legal protection, shielding engineers from accusations of negligence. The following scenario in [9] involving an 8-person company that developed computer-controlled valves for industries such as pharmaceuticals and chemicals illustrates the critical importance of standardized engineering practices, even for VSEs. The customer contracted the company to inspect the requirements using IEEE software engineering standards. However, the developer was unaware of the IEEE 1028 standard, which describes the types of software reviews and procedures required for execution. After installing the new software, the computer-controlled valves malfunctioned, causing damages in the chemical plant and leading to legal action. The court hearing revealed that the supplier could not provide evidence of an inspection according to the standard, resulting in legal and financial consequences. This incident underscores the necessity of adhering to standardized engineering practices to ensure quality, safety, and legal compliance, regardless of the company's size. It also highlights the potential risks and liabilities that can arise from neglecting these standards. In essence, standardized engineering practices can serve as a cornerstone for VSEs, ensuring quality, legal compliance, and economic viability, thereby crucial to their success and sustainability.

2.2 Requirements and Systems Engineering

In Aiwell's endeavour to improve its engineering practices, requirements quality emerged as a pivotal focus area. The importance of requirements quality is multi-fold. First, well-defined requirements serve as the cornerstone for successful project outcomes, reducing the likelihood of costly revisions and delays. Second, they facilitate clear communication among stakeholders, ensuring alignment on objectives and expectations. There is a strong positive link between capability in requirements definition and management [10]. A high-quality requirement is clear, concise, verifiable and traceable. It should be devoid of ambiguity, allowing for a single interpretation, and verifiable through inspection, analysis, or testing. Critically, a requirement should also be traceable, linking back to its source, rationale, and dependencies, which aids in managing changes and understanding impacts.

2.3 Large Language Models (LLMs)

Large Language Models (LLMs) are a subset of artificial intelligence models designed to generate human-like text. These models have gained significant attention in software engineering, particularly in automating coding tasks and improving code quality. LLMs have been evaluated for their effectiveness in code generation and have shown promising results in various aspects of software development [11]. The study revealed that 85% of developers felt more confident in their code quality when authoring with GitHub Copilot. Moreover, code reviews were completed 15% faster, and 88% of developers reported maintaining a flow state, indicating increased focus and reduced frustration. LLMs are not just accelerating the coding process but are also enhancing the quality of the code. GitHub's internal metrics for code quality—readability, reusability, conciseness, maintainability, and resilience—showed significant improvement when developers used GitHub Copilot[12].

Using artificial intelligence, and more recently Deep Learning, to support engineering is not new [13], but in learning to use language, LLMs are also picking up other human skills, such as learning from a single example [14]. Whereas not long ago training an AI was an expensive process that took a long time, LLM foundation models can learn by just looking up the documentation [15]. It is this last newfound ability that Aiwell aimed to exploit in their pilot study.

3 Case Study: Leveraging LLMs to Enhance Requirements Reviews in a VSE

3.1 Case Description

Aiwell is a company with seven employees producing software and hardware used in building automation systems. Due to high workloads, Aiwell found it challenging to allocate sufficient time on a consistent basis to define high-quality requirement specifications for its diverse range of projects. The company wanted to leverage LLM-based tools to automate and enhance the quality of stakeholder and systems requirements, guided by the INCOSE Systems Engineering Handbook [16] and the Systems Engineering Book of Knowledge (SEBoK) [17]. Engineers at Aiwell already used GitHub for version control and wanted to explore GitHub Actions for task automation. They employed ChatGPT Plus and GitHub Copilot for AI-assisted script design and the OpenAI GPT-4 API in script execution. Key metrics included the number of requirement revisions, time spent on each requirement, and the frequency of requirement-tagged issues approved without further revisions. Initial implementation led to more precise and complete requirements, with fewer revisions and reduced time spent on each requirement. This section provides details and practical insights by demonstrating how Aiwell used LLMs effectively and efficiently regarding resources and time to improve their engineering practices, in particular with regards to improving requirements quality.

3.2 Methodology

The study employed two primary methodologies: Action Research and Glaser's Grounded Theory Method (GTM).

Action Research (AR) is a participatory, iterative methodology focused on solving real-world problems through cycles of planning, action, observation, and reflection [18]. Aiwell conducted multiple cycles of Action Research to refine the approach, resolving immediate challenges and planning for long-term adaptability.

Glaser's Grounded Theory Method (GTM) is a qualitative research approach that emphasises the generation of theory directly from data through iterative coding and constant comparative analysis [19]. GTM was chosen for its flexibility in handling qualitative data. The study began by applying open coding to raw electronic communication, including instant messaging and comment threads. This preliminary analysis identified initial concepts, which were continuously refined into categories and themes. The study adopted a constant comparative analysis, integrating new data iteratively to evolve the emergent theory. This approach made sure the resulting theory was both relevant and contextually grounded.

3.3 Ethical and Commercial Considerations

Given Aiwell's status as a Very Small Entity (VSE), the study had to account for ethical concerns associated with data confidentiality and anonymity. Strategies were based on SAGE guidelines [18], and included focusing on collective insights to preserve internal anonymity and working closely with Aiwell to redact sensitive or proprietary data.

3.4 A Working Definition of Requirements Quality

To quantitatively assess the quality of a requirement, Aiwell employed a scoring system based on key attributes such as clarity, conciseness, testability, and traceability. Each attribute was assigned a weight, and all requirements were evaluated on a scale of 1 to 5 for each attribute. Aggregating the attribute scores produced a composite quality score. For example, a requirement with a clarity score of 4, conciseness score of 5, testability score of 3, and traceability score of 4 would yield a composite score of $(4 * 0.3) + (5 * 0.15) + (3 * 0.35) + (4 * 0.20) = 3.8$.

Engineers evaluated requirements using an approach based on Planning Poker [20], an agile estimating technique where team members use playing cards to vote on the complexity of a task, facilitating discussion and consensus.

Requirements engineering best practices were drawn from two seminal resources: "The INCOSE Systems Engineering Handbook" and "The Systems Engineering Book of Knowledge" (SEBoK). These resources were also accessible to the LLMs through hyperlinks, as was a written description of the system context and the scoring system used by human evaluators. This provided a basic framework for assessing requirements.

3.5 Integrating LLMs in the Workflow

Introducing LLMs into the engineering workflow had to be low-cost and require minimal training. ChatGPT and GitHub Copilot served as the primary LLM-based tools for this purpose. ChatGPT is an AI conversational agent capable of generating human-like text based on the prompts it receives. GitHub Copilot is an AI-powered code assistant that helps engineers write new code and understand and work with existing code more efficiently. The tools' affordability made them attractive choices for a budget-conscious small entity. GitHub Copilot's seamless integration with Microsoft Visual Studio Code, a free code editor already used by Aiwell, facilitated a smooth transition. The OpenAI GPT-4 API, though demanding a steeper learning curve, was rendered more acessible with the assistance of ChatGPT and GitHub Copilot. While the code generated by these tools might not always be perfect, it was sufficient to expedite the development processes. LLMs could also participate in multi-lingual, realistic and nuanced human-like discussions with engineers about subtle topics like requirement quality.

Initially, Aiwell engineers started with the default interfaces of ChatGPT and GitHub Copilot. These tools' ease of use and intuitive interface meant that engineers could begin leveraging their capabilities without extensive training or preparation. This factor was crucial, given the resource constraints typical of VSEs. The user-friendly and adaptable tools fit effortlessly into Aiwell's existing GitHub-based version control and task automation workflows.

When OpenAI introduced custom instructions as a new feature in ChatGPT, Aiwell incorporated it into their workflow by adding links to relevant references as standard prefixes to their prompts. This allowed engineers to guide the AI model more effectively, aligning the generated responses with the project's specific context and needs. Standardising custom instruction enhanced the quality of LLM responses by making them more precise and context-appropriate.

As the team gained experience, they began exploring more advanced features of the GPT-4 API. They started crafting prompts that communicated their intent more clearly to the LLMs, improving the quality of the generated response, whether code or requirements evaluation. This incremental and organic developer-led approach to adopting LLM features ensured the team could adapt without feeling overwhelmed, thereby maintaining productivity.

LLMs became an integral part of Aiwell's engineering practices, not as a disruptive technology requiring a steep learning curve but as enablers that were incrementally adopted. The metrics, including the reduced requirement revisions and time spent on each requirement, validated the effectiveness of integrating LLMs into the workflow. Crucially, these gains never required advanced applications such as pre-trained AI models or curated vector databases with custom knowledge but relied only on free and low-cost, publically available tools.

3.6 Collaborating With LLMs on Writing Automation Scripts

Aiwell's engineers leveraged ChatGPT to automate GitHub Actions, focusing on requirements validation. The process began with a simple prompt to ChatGPT, asking it to draft a GitHub Action script to call the GPT-4 API if a user labelled a GitHub repository issue as a requirement. Despite some initial regular expression and JSON parsing challenges, the engineers iteratively refined the prompts, leading to effective scripts. Figure 1 shows an abbreviated example of the generated code.

GPT-4's output, posted as a GitHub comment on the issue, comprehensively evaluated the requirement using SEBoK and the INCOSE Handbook as a reference and provided the engineer with a task list of suggested improvements. Figure 2 shows the full analysis of a requirement generated by GPT-4. Later iterations included a composite score based on the provided guidelines. The AI-generated comment would break down the requirement's clarity by assessing its specificity, conciseness by indicating unnecessary details, testability by evaluating the clearness of acceptance criteria, and traceability by checking its linkage to system needs or stakeholder requirements. Each attribute would receive a score, and GPT-4 would calculate a composite score using the same scoring system and guidelines employed by Aiwell's human engineers. This approach enriched the requirements validation process, offering a quantitative and qualitative assessment to supplement the human scoring and encourage critical discussion. It also showed how VSEs, even with limited resources, can incrementally integrate LLMs into their existing workflows. The use of ChatGPT in script automation not only streamlined the task but also added a layer of intelligence and review, making the process more robust and efficient. This part of the study demonstrated the capability of LLM base models to perform complex systems engineering tasks that align with established systems engineering principles.

```
name: Check Requirement Correctness
on:
  issues:
    types: [labeled]
jobs:
  check-requirement:
    runs-on: ubuntu-latest
    if: contains(github.event.label.name, 'requirement')
    steps:
    - name: Checkout code
      uses: actions/checkout@v2
    ...
    - name: Check  requirement
      env:
        GITHUB_TOKEN: ${{ secrets.MY_GITHUB_TOKEN }}
        OPENAI_API_KEY: ${{ secrets.YOUR_OPENAI_API_KEY }}
      run: |
        ...
        #Load the Statement of Work
        SYSTEM_CONTEXT=$(cat README.md)
        SCORING_GUIDELINES=$(cat scoring.md)

        # Get the content related to checking correctness of system requirements
        CORRECTNESS_GUIDELINES="https://sebokwiki.org/wiki/Stakeholder_Requirements_Definition#Checking_Correctness_of_System_Requirements"
        ...
        # Call GPT-4 API with issue details
        RESPONSE=$(curl https://api.openai.com/v1/chat/completions \
        -H "Content-Type: application/json" \
        -H "Authorization: Bearer $OPENAI_API_KEY" \
        -d '{
            "model": "gpt-4",
            "messages": [
                {"role": "system", "content": "$SYSTEM_CONTEXT $CORRECTNESS_GUIDELINES $SCORING_GUIDELINES" },
                {"role": "user", "content": "Title: $ISSUE_NAME. Id: $ISSUE_NUMBER. Description: $ISSUE_BODY"}
            ]
        }')
        ...
         # Add the response as a comment to the issue using GitHub API
        curl -s -X POST -H "Authorization: token $GITHUB_TOKEN" -H "Content-Type: application/json" -d "$PAYLOAD" "$COMMENT_URL"
```

Fig. 1. Partial GitHub Action yml script generated by ChatGPT

```
The temperature of the glycol must not exceed 50 degrees.

 ## AI Generated Evaluation
 The requirement is clear, concise, and unambiguous. However, it
    could be improved by providing more context and specificity.

While it states a specific temperature limit, it does not specify
    the conditions under which this limit applies (e.g., during
    operation, at rest, etc.). Also, it does not provide any
    acceptance criteria or a method for testing this requirement.

Here's a tasklist for improving this requirement:

- [ ] Specify the conditions under which the temperature limit
    applies. For example, "The temperature of the glycol must not
    exceed 50 degrees Celsius during operation."
- [ ] Define the acceptance criteria. For example, "The system is
    considered to meet this requirement if the glycol temperature
    remains below 50 degrees Celsius during a continuous 24-hour
    operation under standard snowfall conditions."
- [ ] Provide a method for testing this requirement. For example,
    "This requirement can be tested by operating the system
    continuously for 24 hours under standard snowfall conditions
    and monitoring the glycol temperature."

After these improvements, the requirement might look like this:

 The temperature of the glycol must not exceed 50 degrees Celsius
    during operation. The system is considered to meet this
    requirement if the glycol temperature remains below 50 degrees
    Celsius during a continuous 24-hour operation under standard
    snowfall conditions. This requirement can be tested by
    operating the system continuously for 24 hours under standard
    snowfall conditions and monitoring the glycol temperature.
```

Fig. 2. GPT-4 generated evaluation of requirement

3.7 Challenges and solutions

Data Privacy and Security Integrating GitHub CoPilot, ChatGPT, and the GPT-4 API into Aiwell's workflow presented significant security and privacy challenges, primarily due to a lack of clear policies and concerns expressed by government agencies [21]. To mitigate this uncertainty, the tools were initially restricted to non-sensitive material. However, as the study progressed, the introduction of GitHub CoPilot for Business and Microsoft's Azure hosted versions of OpenAI's models provided more secure alternatives [22]. OpenAI's also clarified in its policies that GPT-4 API data is not stored by their servers, alleviating some concerns [23].

Model Limitations While ChatGPT and GPT-4 demonstrated proficiency in domain-specific language, the models faltered in tasks like regular expression parsing, mathematics and JSON formatting. OpenAI's subsequent update, which introduced function-calling capabilities in the GPT-4 API, addressed these issues, enabling deterministic functions for complex tasks.

Human-AI Collaboration Balancing human and AI contributions proved challenging. Although ChatGPT generated effective YML scripts, its numerical requirement scores often diverged from human evaluations. Action Research methodology helped here, as discrepancies triggered discussions, leading to an evolution of the evaluation criteria and a more stable consensus among engineers. Also, when given access to the comments section on GitHub, GPT-4 would join the discussion and be as opinionated as a human engineer if instructed specifically to be so.

Quality Control Ensuring the quality of AI-generated evaluations required vigilant oversight. A systems engineer consistently reviewed the model's outputs, maintaining a human-in-the-loop approach at all times. This review process was integral to the iterative cycles of planning, action, observation, and reflection.

Error Handling Scripts generated by ChatGPT and Girhub Co-Pilot underwent rigorous scrutiny and testing, and engineers provided the tools with prompts based on a template that evolved incrementally based on previous mistakes made by the LLMs. However, since GPT-4's output was limited to comments and not executable code, the risk of operational disruptions was minimised.

Scalability Scaling the approach for larger tasks or teams posed challenges. The solution involved using small, template-based scripts and leveraging LLMs for extensive commenting and documentation.

In summary, the challenges encountered by Aiwell were systematically addressed, often benefiting from the iterative and reflective nature of the Action

Research methodology. This approach not only resolved immediate issues but also contributed to the long-term adaptability and robustness of the engineering workflow.

4 Emergent Themes

The Grounded Theory Method's iterative nature was pivotal in the continuous refinement and evolution of the emergent themes. By ensuring that these themes were deeply rooted in the experiences and feedback of Aiwell's engineers, the study captured the authentic challenges and opportunities of integrating AI into the workflows of a VSE. Several distinct themes emerged from the study, each shedding light on different facets of integrating AI into the workflows of a small company like Aiwell. These themes provide insights into the immediate benefits and challenges of AI adoption and hint at the broader implications for the future of engineering practices.

Accessibility The ease of integrating AI tools into Aiwell's workflows underscored the theme of feasibility and accessibility. Some engineers had initially perceived the adoption of AI as a daunting task. However, the user-friendly nature of tools like ChatGPT and GitHub Copilot facilitated a smooth transition. A comment from an Aiwell team member captured this sentiment: "We thought integrating AI would be a massive undertaking, but these tools made the transition surprisingly smooth." This theme emphasises the democratisation of advanced technologies, making them accessible even to smaller entities. In the early stages of the study, the data pointed towards the feasibility of integrating AI tools into Aiwell's workflows. However, as engineers gained more hands-on experience with these tools, their feedback began to reflect a broader perspective. Comments like "The integration was smoother than we anticipated" highlighted the feasibility and accessibility of advanced AI technologies. This development underscores the importance of user experience in determining the success of technology adoption.

Human-AI Collaboration The interplay between human judgment and AI-generated content emerged as a significant theme. While AI provided valuable insights and suggestions, the human element remained crucial. One engineer observed, "While the AI provides a solid starting point, our team discussions around its suggestions have become invaluable." This theme highlights the symbiotic relationship between humans and AI, where each complements the other's strengths. The initial data suggested a theme centred around the efficiency gains from AI. However, as engineers interacted more with the AI tools, the theme shifted towards collaboration. An engineer's observation, "The AI doesn't replace our judgment but enhances our discussions," was instrumental in this shift. This refined theme emphasises the complementary nature of human intelligence and AI, suggesting that the actual value of AI tools lies in its ability to augment human capabilities rather than replace them.

Productivity and Quality Enhancement The data consistently highlighted the positive impact of AI on both productivity and the quality of work. Engineers reported fewer errors and an increase in overall efficiency. For instance, an engineer at Aiwell noted, "With AI assistance, we've seen a noticeable reduction in errors and an increase in our overall efficiency." This suggests that AI tools can significantly enhance the day-to-day operations of a company, leading to tangible improvements in output quality. The initial data collection phase primarily highlighted productivity metrics, with engineers emphasising AI integration's speed and efficiency gains. However, the theme evolved as the study progressed. An engineer's comment, "It's not just about doing things faster. The quality of our work has improved significantly with AI assistance," underscores the dual impact of AI tools on both productivity and quality. This feedback led to a more comprehensive understanding of this theme, emphasising that AI's value proposition can extend beyond speed to encompass output quality. Notably, this improvement in quality was not primarily due to the quality of the output generated by the LLM, which was sometimes lacking. Rather, the LLM guided the engineer to consider a broader perspective, prompting more detailed requirements and drawing attention to potential edge cases.

Requirements Engineering and Quality Competence The proficiency of AI in enhancing requirements engineering stood out. Engineers found that AI suggestions often prompted them to rethink and refine their initial ideas, leading to clearer and more precise requirements even when they were not using AI tools. An engineer remarked, "Our requirements have never been clearer. The AI's suggestions often prompt us to rethink and refine our initial ideas." This result was distinct from the quality improvement resulting from the guided process. Engineers would gradually adopt thought patterns similar to the LLM-generated output, with the AI essentially serving as a Systems Engineering mentor.

Data Privacy Concerns Data privacy emerged as a pressing concern, especially with tools like GitHub Copilot. Engineers expressed worries about proprietary code being accessed by AI and potentially appearing in someone else's work. An Aiwell engineer expressed this concern, stating, "We need to ensure our proprietary code remains confidential, even when using AI tools." This theme underscores the need for robust data privacy measures when integrating AI into workflows. The broader theme of ethical considerations in AI adoption initially encompassed ethical and data privacy concerns. However, as the study delved deeper into the engineers' experiences, it became evident that data privacy was a more pressing concern. Specific feedback like "Our main worry is our code's confidentiality with AI tools" led to a sharper focus on data privacy, highlighting the need for robust measures to protect proprietary information.

Scalability Challenges While the study showcased the benefits of AI in simple applications, questions arose about scaling these benefits for more complex tasks. Engineers pondered the AI's capabilities in handling intricate processes as the company grows and diversifies. An engineer pondered, "The AI has been great for our current tasks, but how will it handle more complex processes in the future?" This theme points to the need for continuous evaluation and adaptation as companies scale their operations. Emerging later in the study, the theme of scalability challenges captured the engineers' forward-looking perspective. While the immediate benefits of AI in Aiwell's current tasks were evident, engineers began to consider its long-term applicability. One engineer's remark, "The real test will be scaling these AI benefits to our more intricate projects," highlighted the need to evaluate AI's capabilities as companies grow and diversify continuously.

5 Conclusions and Future Work

5.1 Key Findings

This study examined the integration of Large Language Models (LLMs) into the engineering practices of Aiwell, a Very Small Entity (VSE) . The findings underscore the potential of LLMs in enhancing engineering quality, productivity, and overall efficiency within VSEs.

1. **Impact on Productivity and Quality**: When effectively integrated, the study found that LLMs can significantly boost productivity and enhance the quality of engineering practices. This is particularly crucial for VSEs, which often operate under resource constraints.
2. **Feasibility and Accessibility**: Despite their limited resources, VSEs can harness the power of LLMs to address unique challenges and streamline their operations.
3. **Human-AI Synergy**: The study highlighted the complementary nature of human intelligence and AI. LLMs do not replace human judgment but augment it, leading to enriched discussions, better decision-making, and more informed engineering practices.
4. **Ethical and Data Privacy Considerations**: As with any technological integration, ethical and data privacy concerns emerged. The study emphasised the importance of addressing these concerns proactively, ensuring that the integration of LLMs aligns with ethical standards and data privacy regulations.

5.2 Future Work

The findings in this study pave the way for several avenues of future research and exploration:

1. **Expanding the Scope**: While this study focused on a relatively simple technical implementation and narrow topic, future research could explore the impact of LLMs on VSEs in more complex settings. It would be insightful to understand how the challenges and benefits scale with the complexity of

the application since VSE engineers do not have time to become experts in AI application development. What can VSEs achieve without this deep competency?
2. **Long-term Impact**: This study provided a snapshot of the immediate benefits and challenges of LLM integration. A longitudinal study could offer insights into the long-term impact of LLMs on VSEs, tracking their evolution over time.
3. **Customisation and Fine-tuning**: This study used base models and publically available and generic reference materials. Future work could explore the customisation and fine-tuning of LLMs to cater to specific organisational needs, enhancing their effectiveness and relevance. Is this kind of customisation possible using only the resources available to VSEs?

The integration of LLMs into VSEs presents a realm of possibilities, challenges, and opportunities. This is a nascent technology, and the future promises a deeper, more comprehensive exploration of the potential of LLMs in diverse settings and contexts.

References

1. ISO/IEC TR 29110–1:2016: Systems and software engineering-Lifecycle profiles for very small entities (VSEs)-Part 1: Overview, International Organization for Standardization, Geneva (2016)
2. Entrepreneurship at a Glance 2017, OECD (2017). `https://doi.org/10.1787/entrepreneur_aag-2017-en`. `https://www.oecd-ilibrary.org/employment/entrepreneurship-at-a-glance-2017_entrepreneur_aag-2017-en` (visited on 11/10/2021)
3. Research and development in the business enterprise sector. ssb.no, `https://www.ssb.no/en/teknologi-og-innovasjon/statistikker/foun/aar-forelopige/2020-10-27` (visited on 11/23/2020)
4. Innovation in the business enterprise sector. ssb.no, `https://www.ssb.no/en/teknologi-og-innovasjon/statistikker/innov/hvert-2-aar/2019-10-11` (visited on 11/23/2020)
5. Muñoz, M., Mejía, J., Peña, A., Laporte, C.Y., Gasca-Hurtado, G.P.: Beyond factors that motivate the adoption of the ISO/IEC 29110 in Mexico: An exploratory study of the implementation pace of this standard and the benefits observed. IET Software **n/a** (2021). `https://doi.org/10.1049/sfw2.12041`. `https://onlinelibrary.wiley.com/doi/abs/10.1049/sfw2.12041` (visited on 11/10/2021)
6. Larrucea, X., O'Connor, R.V., Colomo-Palacios, R., Laporte, C.Y.: Software Process Improvement in Very Small Organizations. IEEE Software **33**(2), 85–89 (2016). `https://doi.org/10.1109/MS.2016.42`
7. Tran, X.-L.: Systems Engineering Tool Selection Framework for Australian Defence Small and Medium Enterprises. PhD thesis, University of South Australia (2014).
8. Figueiredo, P., Gutenberg, S., Sbragia, R.: Risk-sharing partnerships with suppliers: the case of Embraer. In: Challenges in the Management of new Technologies, pp. 241–262. World Scientific (2007)

9. Laporte, C.Y., Munoz, M.: Not Teaching Software Engineering Standards to Future Software Engineers-Malpractice? Computer **54**(5), 81–88 (2021). https://doi.org/10.1109/MC.2021.3064438. https://ieeexplore.ieee.org/document/9426987/ (visited on 11/10/2021)
10. Elm, J.P., Goldenson, D.R.: The Business Case for Systems Engineering Study: Results of the Systems Engineering Effectiveness Survey: Defense Technical Information Center, Fort Belvoir, VA (2012). https://doi.org/10.21236/ADA610517. http://www.dtic.mil/docs/citations/ADA610517 (visited on 11/21/2021)
11. Ziegler, A., Kalliamvakou, E., Li, X.A., Rice, A., Rifkin, D., Simister, S., Sittampalam, G., Aftandilian, E.: Productivity assessment of neural code completion. In: Proceedings of the 6th ACM SIGPLAN International Symposium on Machine Programming, pp. 21–29. ACM, San Diego CA USA (2022). https://doi.org/10.1145/3520312.3534864. https://dl.acm.org/doi/10.1145/3520312.3534864 (visited on 10/27/2023)
12. Rodriguez, M.: Research: Quantifying GitHub Copilot's impact on code quality. The GitHub Blog, (2023). https://github.blog/2023-10-10-research-quantifying-github-copilots-impact-on-code-quality/ (visited on 10/27/2023)
13. Yang, Y., Xia, X., Lo, D., Grundy, J.: A Survey on Deep Learning for Software Engineering, (2020). arXiv: 2011.14597[cs]. http://arxiv.org/abs/2011.14597 (visited on 09/08/2023).
14. Qin, Y., Hu, S., Lin, Y., Chen, W., Ding, N., Cui, G., Zeng, Z., Huang, Y., Xiao, C., Han, C., Fung, Y.R., Su, Y., Wang, H., Qian, C., Tian, R., Zhu, K., Liang, S., Shen, X., Xu, B., Zhang, Z., Ye, Y., Li, B., Tang, Z., Yi, J., Zhu, Y., Dai, Z., Yan, L., Cong, X., Lu, Y., Zhao, W., Huang, Y., Yan, J., Han, X., Sun, X., Li, D., Phang, J., Yang, C., Wu, T., Ji, H., Liu, Z., Sun, M.: Tool Learning with Foundation Models, (2023). arXiv: 2304.08354[cs]. http://arxiv.org/abs/2304.08354 (visited on 09/08/2023).
15. Hsieh, C.-Y., Chen, S.-A., Li, C.-L., Fujii, Y., Ratner, A., Lee, C.-Y., Krishna, R., Pfister, T.: Tool Documentation Enables Zero-Shot Tool-Usage with Large Language Models, (2023). https://doi.org/10.48550/arXiv.2308.00675. arXiv: 2308.00675[cs]. http://arxiv.org/abs/2308.00675 (visited on 09/08/2023).
16. Walden, D.D., Roedler, G.J., Forsberg, K., Hamelin, R.D., Shortell, T.M., on Systems Engineering, I.C. (eds.) (eds.): *INCOSE Systems engineering handbook: a guide for system life cycle processes and activities.* 4th edition. Hoboken, New Jersey: Wiley, 2015. 1 p. ISBN: 978-1-118-99941-7 978-1-119-01512-3.
17. System (glossary) - SEBoK, https://www.sebokwiki.org/wiki/System_(glossary) (visited on 11/24/2022)
18. Bradbury, H.: The SAGE handbook of action research. SAGE, London (2015)
19. Glaser, B.G.: Basics of grounded theory analysis: Emergence vs forcing. Sociology press (1992)
20. Grenning, J.: Planning Poker or How to avoid analysis paralysis while release planning. Hawthorn Woods: Renaissance Software Consulting **3**, 22–23 (2002)
21. Milmo, D.: Italy's privacy watchdog bans ChatGPT over data breach concerns. The Guardian (2023). https://www.theguardian.com/technology/2023/mar/31/italy-privacy-watchdog-bans-chatgpt-over-data-breach-concerns (visited on 10/27/2023)
22. Data, privacy, and security for Azure OpenAI Service - Azure AI services, (2023). https://learn.microsoft.com/en-us/legal/cognitive-services/openai/data-privacy (visited on 10/27/2023)
23. OpenAI Security and Privacy, https://openai.com/security (visited on 10/27/2023)