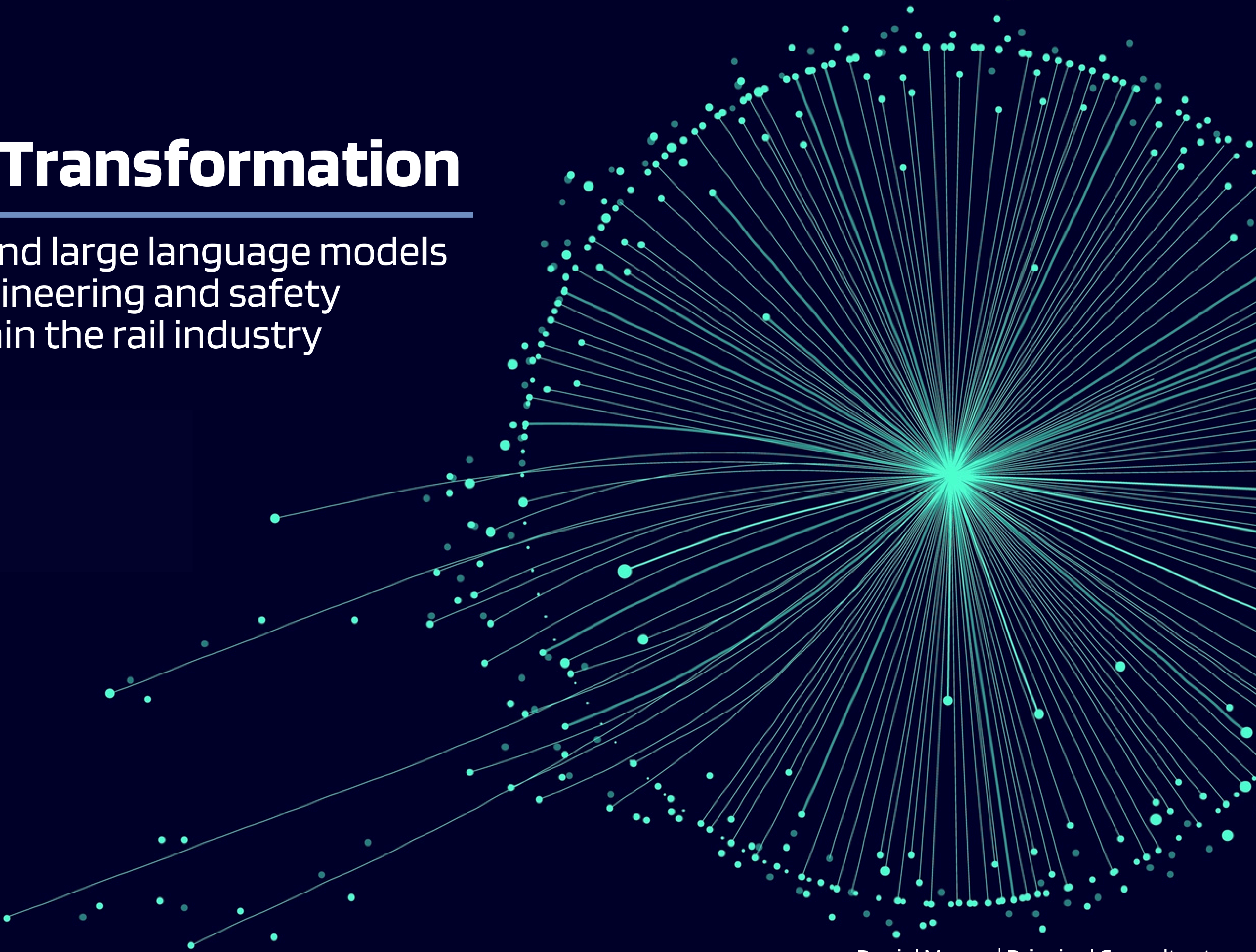
Steering Transformation

The role of AI and large language models in systems engineering and safety assurance within the rail industry



Daniel Munoz | Principal Consultant

Acmena

Insights

As a company that specialises in systems engineering, system safety assurance, and human factors analysis, the core of Acmena's work involves the structured and unstructured evaluation of extensive documentation to execute systems engineering and system assurance lifecycles.

Given the vast volume of documentation consumed and generated in a typical rail project, Acmena has conducted research to explore the potential of leveraging artificial intelligence (AI) to streamline processes and enhance team efficiency with the goal of providing better and more cost-effective services to clients.

In recent years, large language models (LLMs) have gained significant traction across various applications, from text generation to image and video synthesis. LLMs are based on the transformer architecture (Vaswani et al., 2017) and enable the creation of highly capable models through training on massive datasets. During this training process, the weights or parameters within the model are adjusted, allowing the model to develop "intelligent" capabilities.

A key property of LLMs is their generative AI capacity, which encompasses both text generation and a degree of text analysis based on provided instructions.

As Acmena's business revolves around the analysis and production of extensive unstructured data through systems engineering and system safety assurance processes, the potential value proposition of LLMs is of great interest within the industry.

Objectives

The primary objective of the research project included:

- To explore the possibility of creating a tool that would improve documentation drafting efficiency while complying with project and client cybersecurity and privacy requirements.
- Investigate the use of a platform that could be adopted by the company within a budget suitable for small enterprise.
- To assess the potential for the tool to analyse large volumes of unstructured requirements and verification and validation (V&V) data.
- Evaluate the capacity of the tool to rapidly prototype documentation, such as structured arguments, requirement drafting, and system architecture creation.
- Develop a codebase that allows easy access and standardization for deployment into projects, including a standard set of code and techniques for specific functionalities used by teams.
- Explore the possibility for the tool to perform human-like analysis of unstructured documentation.

The Research

Requirements

The research was guided by the following key requirements:

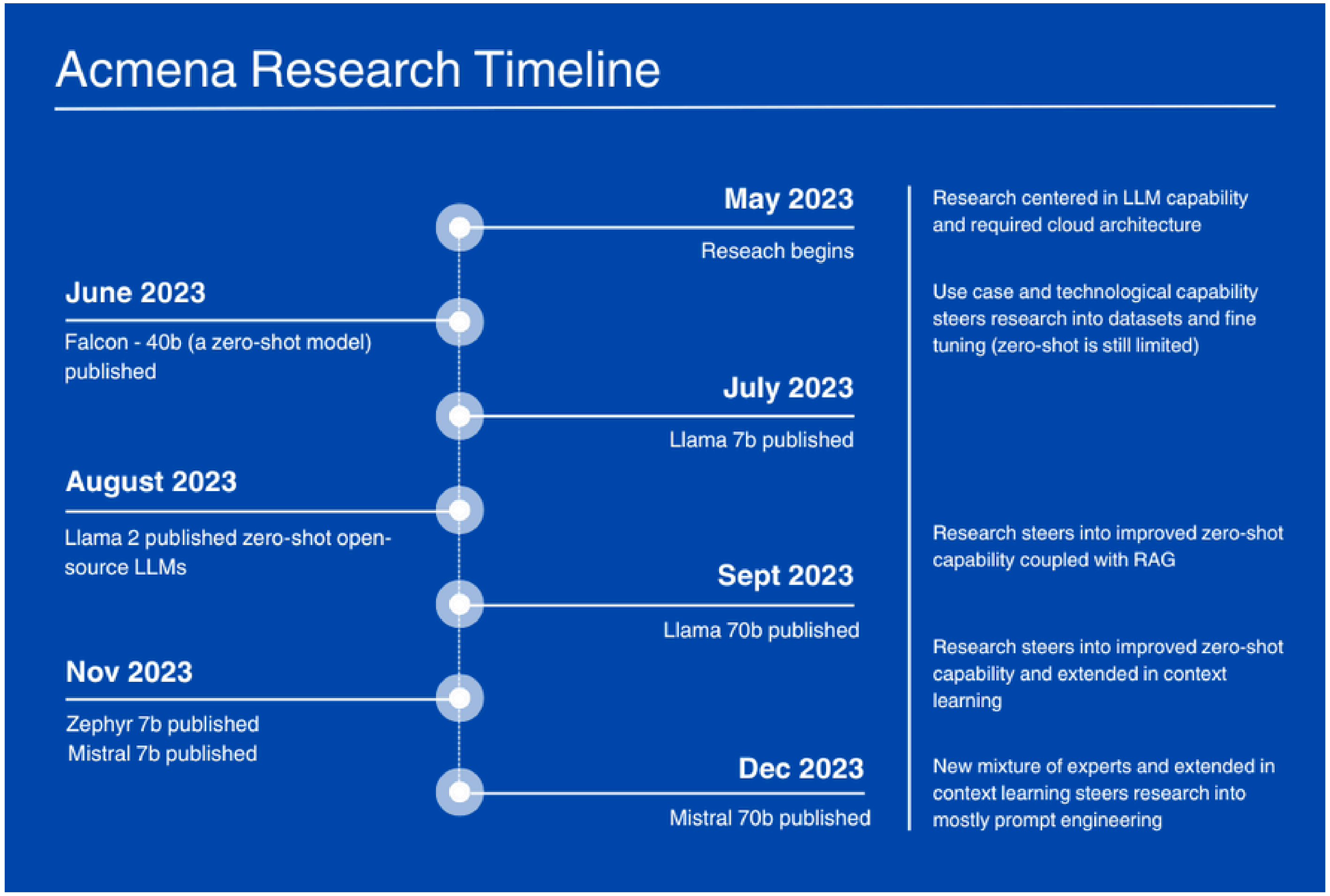
- The solution must be cost-effective for implementation by a small or medium-sized enterprise.
- The solution must achieve a degree of complex reasoning and be capable of both

- consuming and producing large quantitates of text content.
- The solution must comply with corporate data privacy and security policies.

Research Timeline

The research was conducted over a six-month period, from May/June to December 2023. During this time, the progress observed in the industry was overwhelming. Throughout the research, the team had to make decisions due to the technological limitations offered by the open-source LLMs, but breakthroughs occurred almost weekly.

At the beginning of the research, the only available zero-shot LLM (models that are large and complex enough to understand and complete more complex tasks from instructions) was Falcon-40B. To achieve the desired functionality, the team had to research techniques to modify the model's weights through fine-tuning or methods like Retrieval Augmented Generation (RAG). In a simplified explanation, RAG involves using a vector database to provide additional context to the


Acmena Research Timeline
May 2023
Reseach begins
June 2023
Falcon - 40b (a zero-shot model) published
July 2023
Llama 7b published
August 2023
Llama 2 published zero-shot open-source LLMs
Sept 2023
Llama 70b published
Nov 2023
Zephyr 7b published
Mistral 7b published
Dec 2023
Mistral 70b published
Research centered in LLM capability and required cloud architecture
Use case and technological capability steers research into datasets and fine tuning (zero-shot is still limited)
Research steers into improved zero-shot capability coupled with RAG
Research steers into improved zero-shot capability and extended in context learning
New mixture of experts and extended in context learning steers research into mostly prompt engineering

Insights

“Throughout the research, the team had to make decisions due to the technological limitations offered by the open-source LLMs, but breakthroughs occurred almost weekly.”

LLM without needing to modify the model's weights directly. However, these techniques have inherent drawbacks and limitations.

Over the course of the research, multiple and more complex open-source zero-shot models became available, and the context window (the amount of input text the model can process at once) improved significantly. By the end of the research, most of the desired functionality could be achieved using In-Context Learning (ICL) via prompt engineering.

Cost of Implementation
As of 2023, the generative AI space was in high demand with companies like Microsoft and Google competing for resources and paying billions of dollars. In this environment, purchasing hardware capable of hosting a sufficiently complex LLM was challenging and expensive. At the beginning of the research, it was uncertain what model size would achieve the required functionality and what hardware

Insights

requirements such a model would necessitate. However, over the research period it became clear that smaller models (7-13 billion parameters) were useful for text generation but, even when fine-tuned, lacked the capacity to manage tasks requiring more abstract and complex reasoning.

More complex reasoning tasks were better handled by larger and more complex architecture models (mixture of experts and 30-70 billion parameter models). Once this was understood, the hardware requirements for a corporate deployment became much clearer.

Data Privacy Requirements
Given the novelty of the technology and practices, the risk of undetected vulnerabilities, either through the mechanics of the cloud architecture or the structure of the payload used to interact with the LLM, was high.

The team adopted the approach that no public IP addresses would be exposed, and access to the cloud provider would be required to interact with the LLM. During the research period, articles about cybersecurity vulnerabilities, such as "LLM jailbreak" and "Prompt Injection Worms," were published, highlighting the potential for corporate data exposure.

This served as a reminder that new technologies often introduce new vulnerabilities. Until better understood, it is recommended to avoid any public IP access or exposure where data privacy is required. Fortunately, when using a private cloud and prompt engineering, none of the data gets saved in the LLM, and unless intercepted during the prediction process, it gets completely erased from the system.

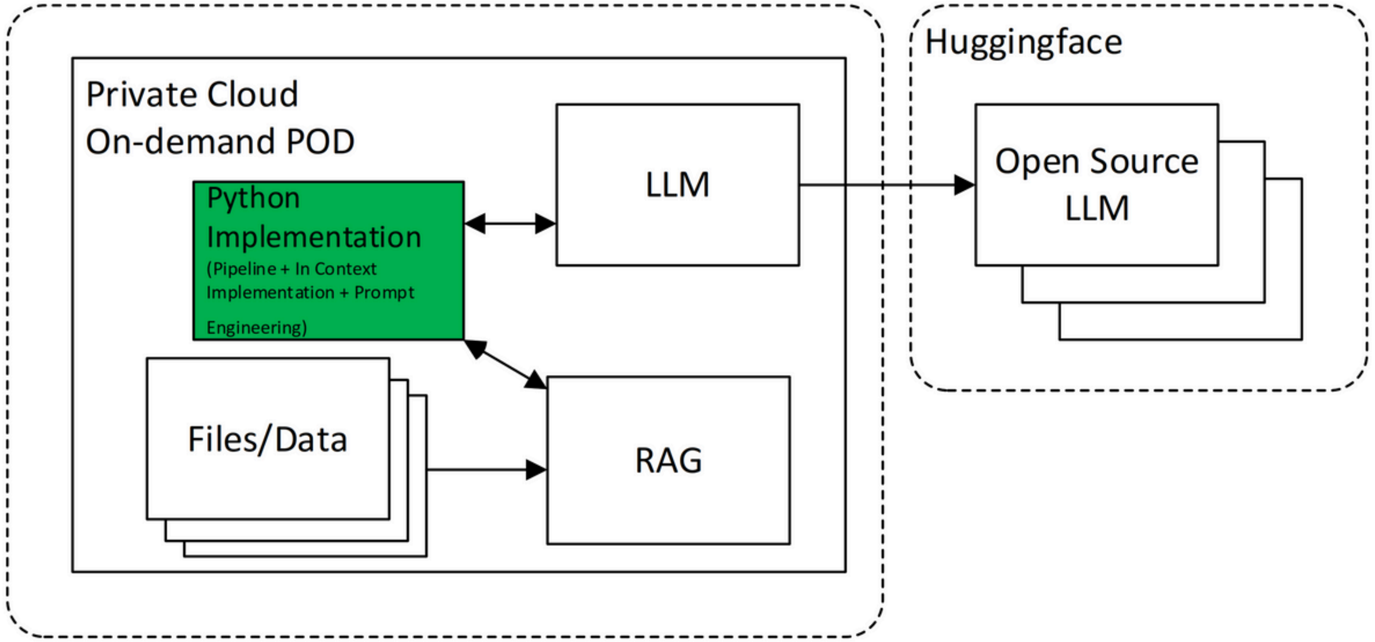
Deployment Architecture
After researching the implementation of a private cloud architecture optimized for LLMs, the simplified deployment considered included a Python web application with a front-end, LLM manager, embeddings manager, and pipeline. This application could load the LLM from an open-source web location or from a local drive. Note that this implementation also allowed for consuming documentation uploaded to a vector database, enabling the use of RAG. The libraries used for the implementation included LangChain and ChromaDB.

Techniques & Evolution During the Project
As mentioned above, there was significant volatility in LLM capabilities during the

Hardware Requirements and Cost

Model Type	Hardware Requirements (GPUs)	Estimated Cost on-premises	Rental Cost
30b-70b Parameters Models (Llama2-70b or Mixtral-7b)	2xH100 80GB GPU	2x40,000 USD	4-15 AUD/hour (approx. 35K-95K AUD/year on 24 hours GPU rental)
7b Parameters Models (Mistral-7b or Zephyr-7b)	1xRTX 6000 GPU	1x7,000 USD	1-3 AUD/hour (approx. 9K-21K AUD/year on 24 hours GPU rental)

Insights



Deployment Architecture Diagram

research period. Initially, the team opted to investigate the use of smaller models and perform task fine-tuning to achieve the desired functionality.

When fine-tuning, the weights of the LLM are modified, and the model "learns" to perform the required task based on the provided data.

Fine-tuning generally requires a substantial investment in computational time to allow the LLM to adjust its weights based on the data, which must be prepared in a question-answer format.

Towards the end of the research, it became clear that the increased reasoning capabilities of open-source models, combined with significantly longer context lengths, allowed for addressing the required functionality using In-Context Learning (ICL) via prompt engineering.

Prompt Engineering & ICL Techniques
One of the conclusions of the research was that the technology available by the end of the research period allowed for achieving the originally envisioned functionality using ICL and incorporating both background data and logic within the prompt.

In prompt engineering, the quality of the prompt is critical to achieving the desired results.

Techniques such as Chain of Thought and Tree of Thought, which direct the LLM towards a logical structure, were found to improve outcomes.

Additionally, XML-like structures when prompting were beneficial in achieving good results.

Insights

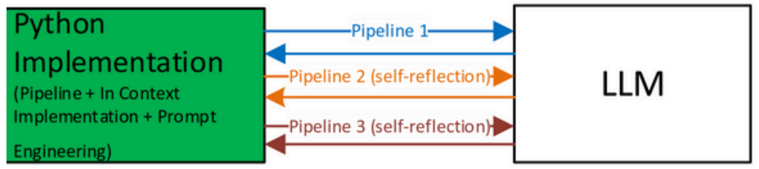
Self-Reflection & Agent-Like Implementation

In addition to prompt engineering, the research explored the possibility of using self-reflection.

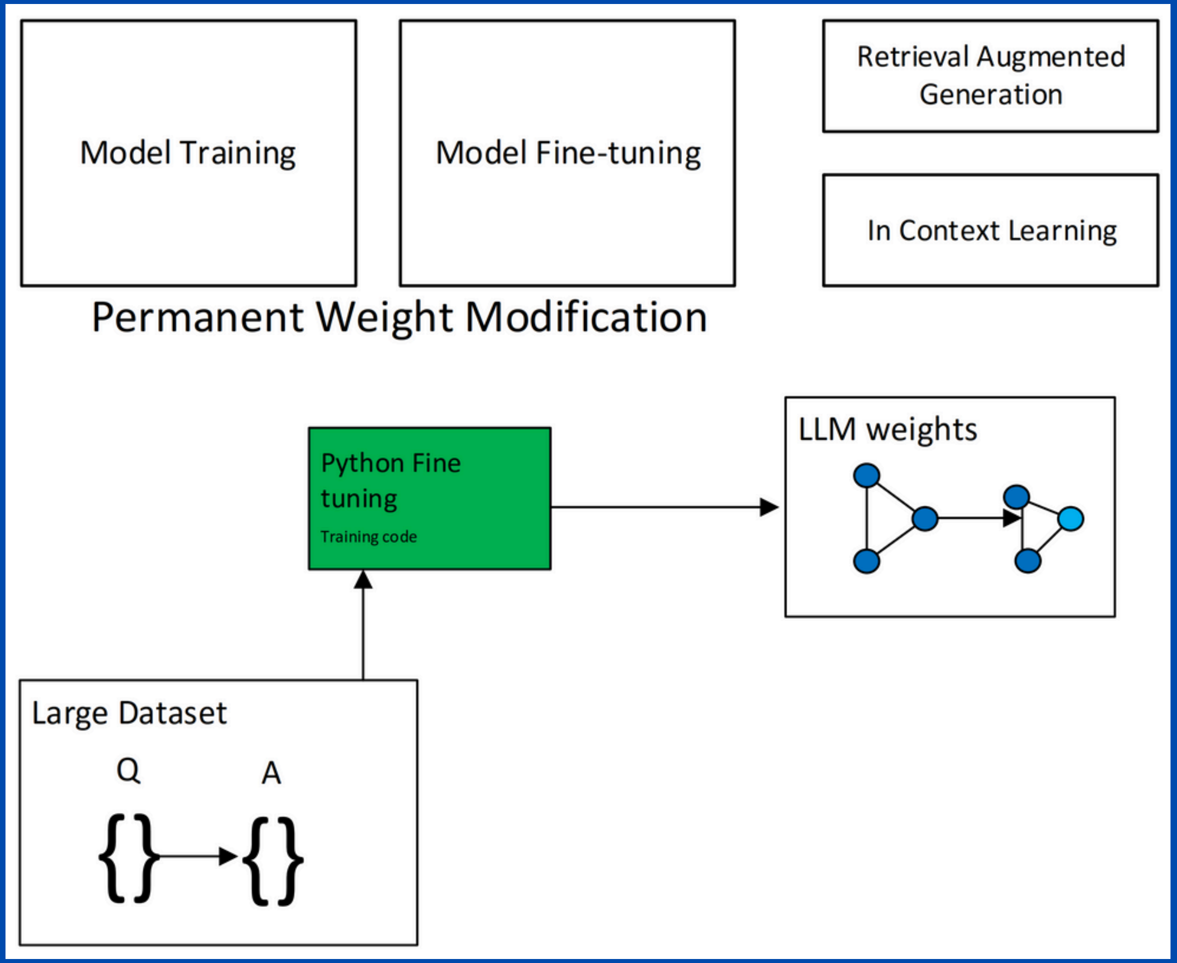
When passing a prompt to the LLM, specific temperature settings (controlling the level of creativity and hallucination risk) are provided. These settings allow the LLM to take more risks in its predictions, which can increase creativity and intelligence but also raises the risk of hallucinations (factually incorrect responses).

The self-reflection implementation allows for incorporating the result of one pipeline into another and provides the possibility to code an algorithm where the LLM checks its own response. The final response is only provided once both the "creator" and "checker" roles confirm the response's correctness.

Integration of Python for LLM Interaction: In-Context Learning, Prompt Engineering, and Self-Reflection Pipelines

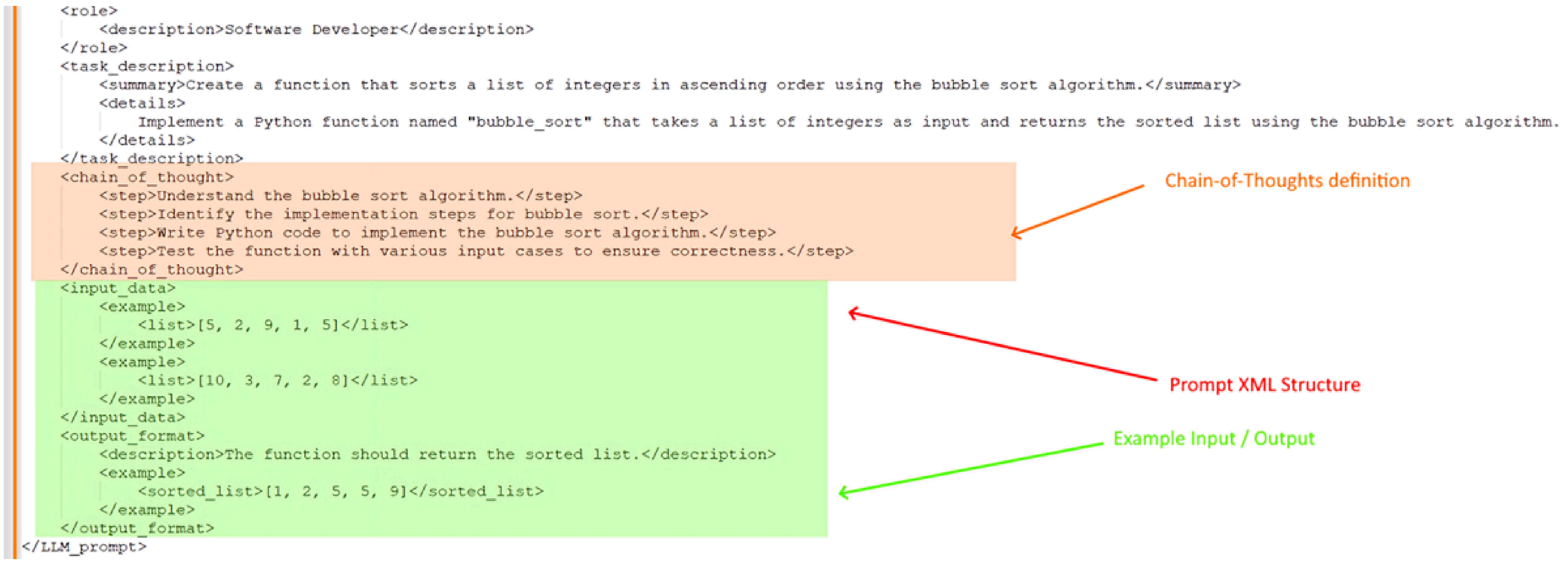


Python Implementation
(Pipeline + In Context Implementation + Prompt Engineering)
Pipeline 1
Pipeline 2 (self-reflection)
Pipeline 3 (self-reflection)
LLM



Overview of Model Training, Fine-tuning and Learning Methods for LLMs

Insights



```
    <role>
        <description>Software Developer</description>
    </role>
    <task_description>
        <summary>Create a function that sorts a list of integers in ascending order using the bubble sort algorithm.</summary>
        <details>
            Implement a Python function named "bubble_sort" that takes a list of integers as input and returns the sorted list using the bubble sort algorithm.
        </details>
    </task_description>
    <chain_of_thought>
        <step>Understand the bubble sort algorithm.</step>
        <step>Identify the implementation steps for bubble sort.</step>
        <step>Write Python code to implement the bubble sort algorithm.</step>
        <step>Test the function with various input cases to ensure correctness.</step>
    </chain_of_thought>
    <input_data>
        <example>
            <list>[5, 2, 9, 1, 5]</list>
        </example>
        <example>
            <list>[10, 3, 7, 2, 8]</list>
        </example>
    </input_data>
    <output_format>
        <description>The function should return the sorted list.</description>
        <example>
            <sorted_list>[1, 2, 5, 5, 9]</sorted_list>
        </example>
    </output_format>
</LLM_prompt>
```

XML Structure for LLM Prompts

Retrieval Augmented Generation
In cases where additional data cannot be injected as part of the prompt, and exists in input documentation, the research investigated uploading PDF documents to vector databases. With this technique the PDF documents are read and the words within are tokenized. The content of the documents is then passed as context as part of the embeddings.

During the research, it was observed that this technique had some potential downsides, as hallucinations were quite common, especially when a large volume of documents was digested and included in the vector database. The team believes that the structuring of the content is critical, and this lack of control over the structure when digesting PDFs could have contributed to suboptimal results and the observed hallucinations.

Insights

Systems Engineering & System Safety Assurance Use Cases Investigated

Systems engineering and system safety assurance often utilize assurance databases such as IBM DOORS to more efficiently manage the lifecycles of technical documentation, including requirement specifications, requirement metadata, traceability, hazards and controls data, and verification and validation evidence.

When well-managed, tools like DOORS provide the possibility to monitor coverage at multiple levels by confirming that links exist at various levels.

These tools include the content for each artifact and the traceability across multiple levels. However, they do not have the capability to analyse the content and support more cognitive analysis, where the content at the multiple levels is interpreted to identify that coverage is achieved not only by having links but by confirming that the content in the lower levels fully supports the content at the higher levels.

Additionally, inconsistencies between levels cannot be automatically checked. Usually, these tasks are performed by V&V engineers during the production of V&V reports. Other tasks, such as drafting structured arguments based on the traceability across multiple levels, are typically done manually by an engineer.

The first area of use case implementation of the deployed platform looked at supporting these time-consuming activities by allowing the ingestion of large volumes of assurance database data and traceability to support the production of drafts. Given the observed issues with hallucinations, this research does not intend for these systems to operate autonomously but rather to empower engineers to focus on value-added tasks and work from advanced generated drafts.

Requirements Traceability, Verification & Validation and Hazard Analysis

When performing requirement traceability analysis, the format of the prompt incorporated a role, a description of the activity indicating the analysis for coverage of requirements at multiple levels, an architecture content, an example of a traceability analysis, the actual traceability to be analysed, and the expected output format.

"During the research, it was observed that the results were not perfect, but the streamlining of what was previously a laborious manual task was worthwhile."

As noted, the prompt is quite large and provides as much detail as possible. While manually constructing such a code template from an exported report from the assurance database is ineffective, it is possible to feed the system entire specifications and review the output.

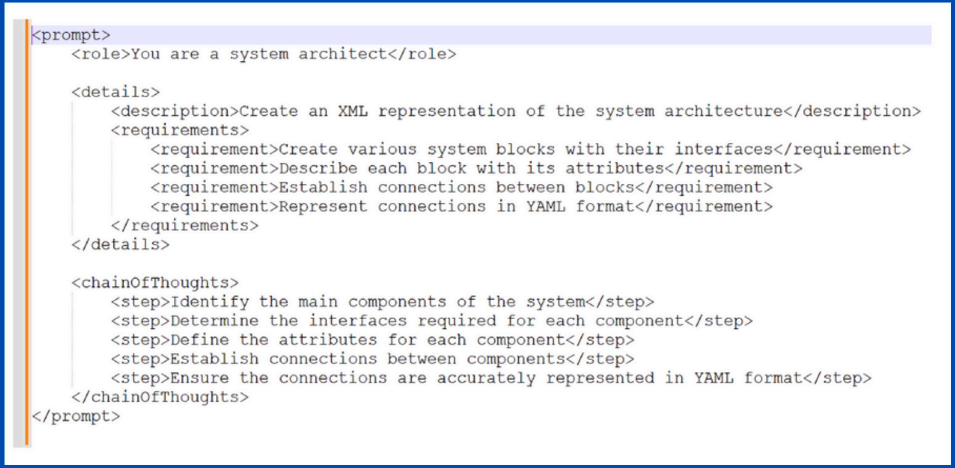
During the research, it was observed that the results were not perfect, but the streamlining of what was previously a laborious manual task was worthwhile. When using more intelligent models such as Mixtral-7b, the results were very promising, with an estimated accuracy of over 80%.

Insights

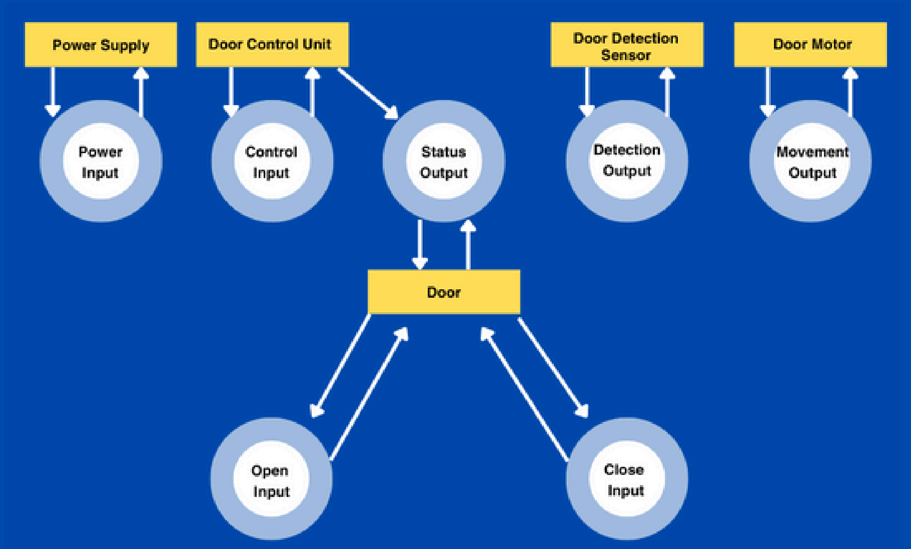
System Architecture Generation

When generating a system architecture, a description of the system and its functions would be provided as part of the prompt, and the expected outcome would be an XML-like structure that includes the different parts of the architecture and interfaces as a rapid prototyped model. Again, the level of detail and context of use is critical for better outcomes, but the rapid prototyping has been used to provide diagrams for discussion and to evolve further based on engineering design discussions. By embracing the use of code-based and XML-like architecture definitions, the LLM can also be used to make modifications over previous versions and therefore streamline changes.

The research found that the system architectures generated by the LLM were almost sufficient for the purpose of project use and discussions, and the fact that the architecture was defined in code meant that a greater degree of change control and change comparison was possible when comparing evolving versions.



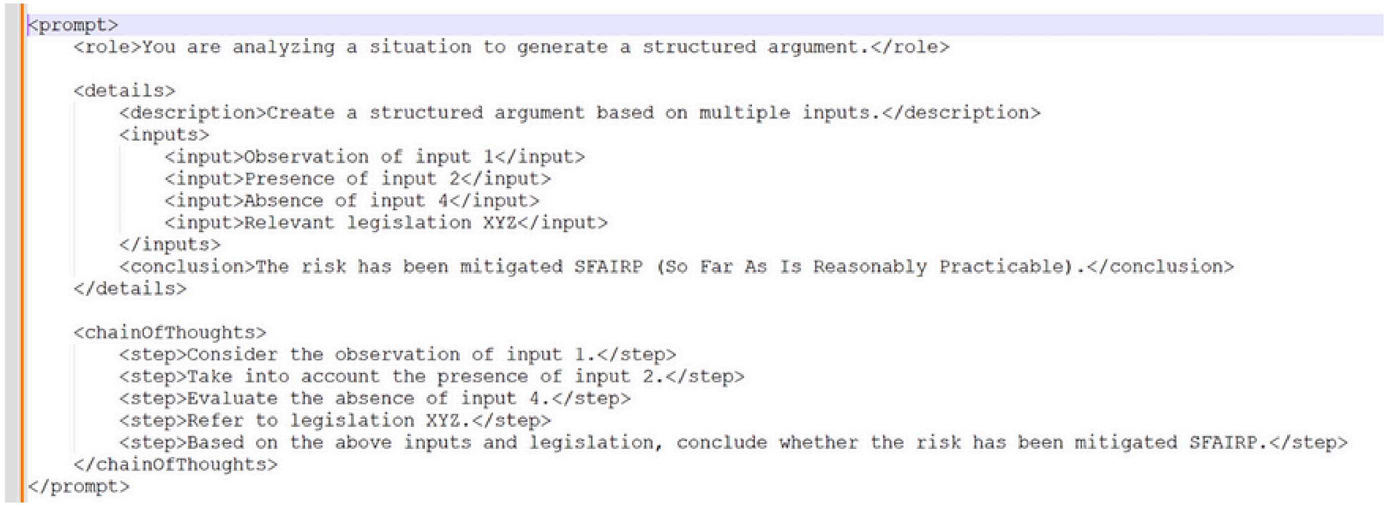
```
<prompt>
    <role>You are a system architect</role>

    <details>
        <description>Create an XML representation of the system architecture</description>
        <requirements>
            <requirement>Create various system blocks with their interfaces</requirement>
            <requirement>Describe each block with its attributes</requirement>
            <requirement>Establish connections between blocks</requirement>
            <requirement>Represent connections in YAML format</requirement>
        </requirements>
    </details>

    <chainOfThoughts>
        <step>Identify the main components of the system</step>
        <step>Determine the interfaces required for each component</step>
        <step>Define the attributes for each component</step>
        <step>Establish connections between components</step>
        <step>Ensure the connections are accurately represented in YAML format</step>
    </chainOfThoughts>
</prompt>
```



Insights

Structured Argument Generation
Structured arguments often rely on multiple different parts of the argument to provide a conclusion that feeds into an overarching argumentation that draws a final conclusion.

For example, a "So Far As Reasonably Practicable" argument will need to consider all the parts of the implementation of multiple requirements and aspects of the architecture implemented to provide a supporting argument. In the research, especially when providing a number of examples, it was possible to generate reasonably good draft arguments.



```
<prompt>
    <role>You are analyzing a situation to generate a structured argument.</role>

    <details>
        <description>Create a structured argument based on multiple inputs.</description>
        <inputs>
            <input>Observation of input 1</input>
            <input>Presence of input 2</input>
            <input>Absence of input 4</input>
            <input>Relevant legislation XYZ</input>
        </inputs>
        <conclusion>The risk has been mitigated SFAIRP (So Far As Is Reasonably Practicable).</conclusion>
    </details>

    <chainOfThoughts>
        <step>Consider the observation of input 1.</step>
        <step>Take into account the presence of input 2.</step>
        <step>Evaluate the absence of input 4.</step>
        <step>Refer to legislation XYZ.</step>
        <step>Based on the above inputs and legislation, conclude whether the risk has been mitigated SFAIRP.</step>
    </chainOfThoughts>
</prompt>
```

Retrieval Augmented Generation
In the research, the team looked at performing a simple request to identify the referenced standards in many equipment datasheets.

This is usually a repetitive task in projects where an assurance engineer is required to identify and confirm that procured equipment is claiming compliance or holds certification to required standards. Such a task is repetitive, laborious, and time-consuming, but the input data is highly unstructured, as each datasheet would contain tables, references, and sections that would not follow a narrative. When researching this task, it was observed that the LLMs would struggle significantly, and hallucinations were common.

This was the only part of the late research where the results were clearly misleading and added negative value to a team using the tool. What became clear is that although the use of an LLM had demonstrated valuable capabilities in input data analysis and content creation, the level of structuring and guidance given as input is critical for the success of the approach.

Conclusions

Objective Completion Analysis
At the end of the research, the initial objectives were achieved as follows:

- Explore the possibility of creating a tool that would improve the efficiency in documentation drafting in a manner compliant with project and client

Insights

- cybersecurity and privacy requirements [ACHIEVED]
- Explore the use of a platform that could be used by the company within a budget affordable by a small company [ACHIEVED]
- Explore the possibility of the tool incorporating the capacity to analyse large volumes of requirement and V&V unstructured data [ACHIEVED]
- Explore the possibility of a tool which has the capacity to rapidly prototype documentation such as structured argument creation, requirement drafting, and system architecture creation [ACHIEVED]
- Create a codebase that allows easy access to team members standardization for deployment into projects, including a standard set of code and techniques for specific functionalities being used by the teams [ACHIEVED]
- Explore the possibility for the tool to perform human-like analysis of unstructured documentation [NOT ACHIEVED]

From the results of the research, we concluded that the technology is ready to support, within a reasonable budget, small and medium-sized businesses with data privacy requirements and documentation-driven deliverables (like consultancies).

The benefits and improvements in productivity are already tangible, which also leads to more time spent focusing on the analysis of safety and likely to improve the level of service to achieve better outcomes. In the case of Acmena, it translates to better outcomes from systems engineering and system safety assurance for clients and infrastructure projects. However, the technology cannot be trusted on its own. It still lacks the capacity to articulate complex, unrelated, multi-step logic, and it is exposed to hallucinations, especially when unstructured input data and instructions are provided.

Overall, the outcome of the research shows significant benefits in using the technology to improve systems engineering and system safety assurance activities.

Further Research

This research paper has focused on the development of a platform suitable for project use and in providing support for content generation.

As observed in the prompt structuring to achieve the desired outcome, it is possible to use prompt engineering techniques to achieve a further complexity in the outcome of the Large Language Models.

Large Language Models have not been great in mathematical calculations in the past but a possible area of further research is to incorporate quantitative and statistical methods as part of chain-of-thoughts in order to integrate mathematical calculations with the LLMs.

This research could potentially support further the system engineering and system safety assurance disciplines by incorporating more complex intelligence and analysis capacity to the models.

Acmena

Contact Us

+61 (0) 478 814 324
enquiries@acmena.com.au
www.acmena.com.au

Acmena Group Pty Ltd
PO BOX 220
Ashgrove West
QLD 4060
ABN: 37 158 514955
ACN: 158 514 955