EngineTalk: supported maintenance of diesel engines aboard navy ships through a RAG enabled LLM system

J.S. Maas MSc [a], B. Smit MSc [a], LtCdr Y. Linden MSc [b*]

[a] CGI Nederland BV, Defense Department
[b] Royal Netherlands Navy, COMMIT (Command Materiel and Information Technology), NLD

* Corresponding Author. Email: Y.Linden@mindef.nl

Synopsis

Over the past years, Large Language Models (LLMs) have matured to a level wherein adoption for naval applications is eligible for serious consideration. In this paper, we describe the implementation of an LLM-driven AI agent for the support of engine maintenance crew in diagnosing and finding solutions to engineering problems. By implementing a RAG component, in combination with various prompt engineering techniques, early results show that we can save significant time and resources for engine operators. Additionally, the implementation of a comprehensive evaluation framework enables avenues for reliable performance indicators.

Keywords: Large Language Model; LLM; Retrieval Augmented Generation; RAG; Propulsion; Maintenance; Usability research; Design Thinking; SAP; Integration aboard; Automation; Marine systems; Navy vessels;

1. Introduction:

In recent years, recruiting and retaining technical personnel has become increasingly challenging for the Royal Netherlands Navy (RNLN), mirroring a broader trend observed across the Dutch labour market. Interest in technical education, particularly vocational pathways that lead to hands-on engineering roles, continues to decline, even as the demand for technically skilled personnel rises across sectors such as construction, civil infrastructure, and the energy transition. At the same time, naval engineering roles are becoming more demanding: onboard systems are growing increasingly complex, while crew sizes are being reduced. As a result, individual engineers are expected to take on broader responsibilities, often requiring them to operate, maintain, and troubleshoot multiple interdependent systems (Tiddens *et al.* 2024) for which support is needed.

To address the persistent shortage of technical personnel, the RNLN has improved in-house technical education programs and increasingly turned to smart automation and advanced support systems. A key focus has been on the digitalization of procedures to reduce the need for direct communication and ensure that critical information is accessible wherever and whenever it is needed. For example, tools such as the Battle Damage Repair Support Tool (van Zijl *et al.* 2024) and D-SACT (Voûte *et al.* 2024) provide structured guidance for technical interventions. However, there remains a significant gap in the operational support available to naval engine maintenance crews when dealing with complex and unfamiliar engineering tasks. As onboard systems become more sophisticated and crew expertise becomes more generalized, maintaining and troubleshooting these systems places increasing cognitive demands on individual engineers. While digital tools have improved access to procedural information, the application of AI technologies, particularly LLMs to provide contextual, task-specific support during complex maintenance activities remains largely unexplored in this domain.

This paper investigates the development and implementation of EngineTalk: a Retrieval Augmented Generation (RAG) enabled LLM designed to support engine maintenance crews on RNLN vessels. EngineTalk assists engineers by answering technical questions, supporting fault diagnosis, and retrieving and combining information from multiple sources, such as maintenance manuals and historical maintenance records extracted from SAP, the RNLN enterprise resource planning software suite. Fully hosted within Dutch Defence infrastructure, EngineTalk is being prepared for deployment aboard ships to ensure functionality in communication-deprived settings. Developed in just six months using agile and design thinking methodologies, a minimal viable product will have been deployed at time of publication. Although this paper does not include operational usage data, it presents insights from user evaluation sessions and technical assessments conducted during development. The findings suggest that such AI-supported tools can significantly enhance fault diagnosis, reduce reliance on shore-based expertise, and improve the operational performance of junior technical crews. This work supports the broader transformation of maintenance practices in naval propulsion, power generation, and distribution systems.

Authors' Biographies:
Jonathan Maas is an AI consultant, trainer, and practice lead at CGI, Netherlands. He has a background in Data Science, Linguistics, and AI.
Bart-Peter Smit is a specialist in Situational Awareness at CGI, Netherlands. He has a leading role in multiple projects focused on reduced manning and data-driven decision-making within the defense and public safety domain.
Youri Linden is a PhD candidate at Delft University of Technology and senior system engineer automation at COMMIT (Command Materiel and Information Technology) in Utrecht, NLD

2. Method:

2.1. Functional Method

To ensure the system would function effectively in the operational environment of a navy vessel, RNLN first defined a clear set of technical requirements. Among these was the ability to store and process at least 1,200 pages of maintenance manuals. Additionally, the system needed to access SAP maintenance data specific to the combined propulsion systems of one class of naval ships, covering a fleet of four vessels. These baseline requirements provided the foundation for the system's technical scope. However, the RNLN recognized that compliance alone would not guarantee practical usability.

To bridge this gap, the project adopted a human-centered design approach. A User Experience designer and a business consultant with expertise in situational awareness and human factors were added to the technical team. Using the Design Thinking methodology (Brown 2009), the team focused on identifying real user needs through a structured, iterative process. This methodology emphasizes empathy, problem definition, ideation, prototyping, and testing. It supports innovation by repeatedly refining the solution based on user interaction.

User engagement played a central role throughout development. Workshops, targeted interviews, and feedback sessions were conducted with shipboard personnel and technical crews. The team visited ships to observe current work processes, identify inefficiencies, and understand points of frustration. In addition, operational engineering experts, who support crews remotely, were interviewed to provide further context and validation. These activities ensured that the solution would align with real-world practices, not just theoretical needs by applying architectural modelling techniques to validate requirements against actual operational use-cases while reducing cognitive bias (Liedtka 2015).

This dual-track approach aimed to create a system that works for its users, not just one that meets its specifications. Close interaction with end-users enabled rapid feedback loops, resulting in an iterative development process. These cycles helped refine both the interface and the system logic. The collected feedback also contributed to the data used in this paper, particularly regarding design decisions and perceived usability. As a result, the system evolved in direct response to operational needs and user insights.

2.2. Technical Method

In this section, we outline the technical implementation of our solution, highlighting some of the heuristics tailored to enhance (system) performance. Due to security considerations, certain implementation details cannot be disclosed in full.

2.2.1. Large Language Model

We utilize a deployment of an open source LLM model, which has been optimized through quantisation methods, and other improvements to enhance token generation speed. As such, the model is supplied as a pair of endpoints, which allows both text-response generation, as well as a text-to-embeddings conversion utility. We utilize text-embeddings with a vector-length of 8192 dimensions.

2.2.2. Prompt Tuning

Proper deployment of LLM-based applications necessitates the configuration of a model's prompt instruction template. Such a template can be used to specify the task a model is expected to perform, in addition to guidelines which the model must adhere to, as well as prompt-engineering techniques to improve the quality of model outputs. For our application, we tailor our prompt instructions as such to provide interactive output, while minimizing the expenditure for the context window. Overall, the agent is instructed to[1]:

1. Act as an helpful AI agent in the support of naval diesel engine maintenance.
2. Take into account what information the end-users have access to, under what conditions they may be operating, and what information the end-users would prefer to utilize.
3. Adhere to style and response preferences, described as a preference for "concrete and actionable, rather than elaborate" responses, in addition to a professional, friendly, and task-oriented tone.
4. Take into account what sources of information the LLM-agent might be supplied by the RAG component.
5. Ensure a generated answer properly addresses user queries, and are also well-grounded in the retrieved context information.
6. Emphasize re-utilization of named entities or codes featured in user queries.

[1] This list is not exhaustive, and provided for illustrative purposes, as providing the actual prompt in this paper may introduce security risks.

7. Engage the user in dialogue, to retrieve additional context, if initial user queries were unclear.
8. Refrain from profanity, as well as kindly refuse from engaging the user in topics not related to the task at hand.

This prompt instruction template was iteratively established, based on user feedback and testing, in addition to stakeholder incentives, and overall best-practices. Though due to context window size constraints, prompt engineering techniques whose effectiveness relies on substantial token-windows were excluded to prioritize clarity and performance.

2.2.3. Retrieval Augmented Generation (RAG)

Most contemporary LLM-driven applications are deployed with the addition of a Retrieval-Augmented Generation (RAG) component. Using a RAG component enables the LLM model to generate its answers based on information derived from unstructured datasets, or documents (Lewis *et al.* 2021).

Intuitively, rather than allow an LLM to directly generate answers to user queries, instead the user query is first converted into a semantic vector embedding. This vector embedding is then used to retrieve partitioned sections of various documents, which are related to the user query. After such relevant text chunks are identified, these are supplied to the LLM model in conjunction with the original user query, as such that the LLM model can effectively utilize this retrieved context to augment its generated response, and ensure that the response is also grounded in well-established prior knowledge.

Our RAG component was implemented using a combination of the OpenSearch and LangChain frameworks, which provide the desired flexibilities for ease of use and implementation of several heuristics. We encapsulate text chunks derived from a selection of several technical engineering manuals and engine maintenance guides, in addition to historical maintenance data derived from SAP records.

2.2.4. SAP integration

SAP is one of the world's leading frameworks used in many industries for Enterprise Resource Planning (ERP). In our specific context, the maintenance module is of particular importance. The recorded history of all maintenance activities, both corrective and preventive, along with the preventive maintenance schedule, provides valuable insights. This repository of data can be highly relevant for addressing specific maintenance tasks, as it may contain solutions to previously encountered, and potentially recurring, issues.

An extraction of records from SAP, containing a variety of relevant fields, is first created to supply content to our application. This step enables the transfer of structured data needed for further analysis and use within our system. The extracted fields provide information on the ship, its installations, and the individual components within those systems. This includes data on operational status, such as running hours and the progress of preventive maintenance, as well as records of faults and corrective maintenance actions. Additionally, the data covers maintenance tasks requested by the ship for execution by the shore-based maintenance crew. In cases where the crew requires assistance with complex faults, the extraction also includes engineering reports documenting support requests and corresponding recommendations from senior shore-based engineers. By treating this descriptive text as a stand-alone data point, we can convert these entries into text embeddings and load the historical records into our RAG vector store as if they were conventional text chunks. To support future retrieval, we also include relevant metadata from each record, which may play an important role in identifying contextually appropriate chunks. This approach allows EngineTalk to respond to user queries by drawing on a combination of technical data from manuals and pertinent historical maintenance records.

In total, we processed a select set of engineering manuals that were determined most relevant for use, spanning a multitude of thousands of pages in total. Additionally, we also extracted several thousands of historical SAP records pertaining to propulsion engine maintenance, spanning approximately a decade of service records.

2.2.5. Hybrid Search

Overall, we utilize a hybrid search approach to chunk retrieval. In this approach relevant chunks are identified not only by virtue of their semantic vector similarity to the user query, but also by a weighted word-matching. Doing so ensures that our chunk-retrieval approach is more resistant to potential inaptitude of a semantic embedding to properly encapsulate domain-specific terminology. This is especially helpful for the retrieval of relevant chunks which feature Dutch text and domain-specific terminology, as most available embedding models are expected to produce notably poorer semantic embeddings for the more obscure languages[2]. Moreover, this is

[2] See https://ai.gopubby.com/why-embedding-matters-when-building-a-non-english-rag-system-multilingual-embeddings-1e3434ea6180 for a further explanation on this topic.

done in a fashion which discounts stop-words, and assigns more weight to key terms. As such, retrieval is improved based on the word-matching of words that are actually unique[3].

2.2.6. Language Invariant Chunk Retrieval

As we anticipated a discrepancy between the language the user-queries were posed in (Dutch), and the language used in our RAG base (a combination of English and Dutch), there was a need to introduce a level of language invariance in the chunk-retrieval process. Indeed, early testing confirmed this hypothesis, and not only were the model embeddings not particularly language-invariant. On the contrary, there was a strong bias to retrieving chunks written in the corresponding language as the user query (even though more relevant chunks were available in English). To counter this, we've implemented a multi-query translation & reformulation technique. As such, a Dutch user-query is first translated to its English equivalent (and vice-versa), and from both the original and translated query, reformulations are generated as well. This results in a multitude of queries, which properly encapsulates a scope of languages and various formulations of a query, and retrieves a collection of multiple chunks, each with their own similarity score. A final selection of chunks is made based on a combination of majority vote (i.e: chunks retrieved by various searches are more likely to be considered for augmenting the output) and the hybrid similarity score.

2.2.7. Dynamic Similarity Score Thresholding

When retrieving relevant chunks to answer a query, it may occur that none of the top-k retrieved chunks suffice for generating an adequate response. Moreover, when only a subset of retrieved chunks suffice for properly answering a query, it would be inefficient to consider additional chunks, if offering these as additional context does not result in a substantially improved answer.
For this reason, we decided to improve chunk-retrieval by establishing a minimum retrieval threshold, to ensure that retrieved chunks abide by some bare minimum relevancy to a user query. That way, rather than having the system retrieve the top-K chunks at all times (and forcing the LLM to spend computation on it), we only retrieve a variable number of chunks, neglecting those that can be pre-emptively determined to not be relevant enough.
Moreover, we extended our implementation to adopt a variable retrieval threshold, demanding that retrieved chunks not only meet some minimum threshold, but also meet another variable threshold, which is a proportion based of the similarity score of the most relevant retrieved chunk. Such an implementation ensures that, when at least one relevant chunk could be identified, we demand of the other retrieved chunks that they not only meet a minimum retrieval threshold, but are at least half as relevant for the query, as the most relevant retrieved chunk is.

2.2.8. Meta-data weighted retrieval scoring

Though our hybrid retrieval heuristic already optimizes retrieval of chunks based on both semantic and syntactic features, we anticipated a necessity to distinguish explicitly between various sources. As such, for each chunk, we further record metadata, including the title of the source wherein a chunk originates from, the corresponding page, and a date of publication. In this fashion, the computation of a chunk's relevance score can be weighted dynamically to prefer some degree of recency, as well as some degree of source. For example, if a user query explicitly mentions keywords that can be linked to the summary of a specific document, then the retrieval score for chunks from this document will be slightly increased. Similarly, this summary metadata is also used to indicate chunks obtained from SAP-data correspond to maintenance records, and can feature important aspect of the record. By doing so, if a user query explicitly features a preference to have its query answered by consulting input from maintenance records, the retrieval score of chunks corresponding to this source is increased as well, ensuring that content from the right sources are retrieved, rather than only the semantically most similar ones.

[3] After all, naïve word-matching would otherwise commence, and match low-content words that do not truly contribute to the relevance of a chunk with a user query.

2.3. Evaluation

Though adoption of LLM (and related) technologies have grown over the past years, evaluation methods of these systems leaves plenty to be desired. Unlike conventional machine learning algorithms, any well-grounded automated evaluation of LLMs quickly departs from classic notions of precision or recall performance-rates.

Rather, we now need to cope with an attempt to quantify both minute semantic and factual differences between generated and reference outputs. While it might initially be tempting to directly quantify such differences through cosine similarity distance between semantic vector embeddings of generated and reference response, such an approach inevitably suffers from the same risks the LLM already does. After all, were such an approach to be considered with a small model, its relatively low resolution in latent space may result in deceptively warped cosine similarity metrics. As shown in (Steck *et al.* 2024), cosine similarity on semantic vectors cannot elegantly guarantee to accurately reflect subtle semantic differences.

Contemporary popular evaluation metrics are the ROUGE and BLEU scores. Though these metrics have better foundations than naïve cosine similarity computation, and can suffice in practice for various tasks (mainly translation), they still rely on assumptions that don't always go without violation (as shown in (Graham 2015) and (Reiter 2018)). Therefore, we instead consider alternatives offered by the RAGAS framework, which features various LLM-driven metrics which penalize semantic, rather than syntactic, differences.

Evaluation Metric	Description	Relevance for our scope
Context Precision	Measures the proportion of context chunks retrieved, which were actually relevant to the query.	Limited. Though evaluating retrieval relevance is valuable, proper execution demands significant resources.
Context Recall	Measures the proportion of relevant retrievable context chunks that were successfully identified and retrieved.	Limited. Though evaluating retrieval coverage is valuable, proper execution demands significant resources.
Noise Sensitivity	Indicates how often the LLM will incorporate irrelevant context, resulting in incorrect or irrelevant answers.	High, as it evaluates robustness and invariance to noise.
Response Relevancy	Measures to what degree a user's query is actually properly addressed by a generated answer.	High, as it assesses answer relevancy.
Faithfulness	Measures the factual consistency of an answer with respect to the provided context or evidence. An answer is faithful if all its claims can be supported by its retrieved context.	High, as it measures whether the LLM properly adheres to grounded information provided by RAG, and does not hallucinate.
Factual Correctness	Gauges the degree wherein a model's answer is factually correct to a reference ground-truth answer.	Limited, as it relies on establishing a ground-truth reference set at considerable costs.

Table 1: An overview of RAGAS evaluation metrics of relevance for our scope. See (Es *et al.* 2025) for in-depth explanations.

Though we pose this evaluation framework for review, proper application of it demands significant efforts in establishing and validating evaluation data. Therefore, within the current scope, we focus primarily on a functional system evaluation. This is conducted instead through a series of functional end-users feedback sessions, wherein we monitor system interaction, and manually report on these to further improve the system performance in line with end-user requirements.

2.4. Ethical Considerations

Just like any other AI solution, ours warrants careful consideration of ethics and data privacy concerns. To address these, we conducted a Responsible AI assessment, and ensured careful monitoring of potential violations. As such, for the current scope we refrain from explicitly recording any additional user-details without their explicit consent. The SAP extract was additionally anonymised to reduce the risk that it might feature personal data, and we ensured to have EngineTalk feature proper disclaimers to dissuade misuse of the LLM system, and stimulate cross-checking outputs with original sources.

3. Results & Discussion

Despite the development of a comprehensive evaluation framework, gathering representative evaluation data for such an assessment proved challenging, and was subject to various logistical constraints. Instead, our solution was evaluated independently through various user-feedback sessions, wherein user experiences were recorded and assessed. Hence, in this section we will cover both user experience and technical results, as well as a discussion of these.

3.1. Functional results

The design thinking methodology provided insights in the operational contexts of the navy crews, influencing design choices and tailoring model results. These user feedback sessions delivered the following insights:

Finding the right information at the right time is a challenge and this system proposes a solution, especially in an operational environment. If a system fails, the maintenance crew must sift through documentation comprising thousands of pages to diagnose and restore the system. This can be rather tedious, especially for crews that are inexperienced with this specific type of ship, with this specific propulsion configuration. The system has been assessed to successfully help in quickly identifying relevant pieces of documentation, paraphrasing the raw information while also linking to the source documentation.

Furthermore, searching in SAP is problematic through a low-bandwidth connection. This makes it difficult for crews to search in the system, therefore relying more on shore support through engineering requests. Shore support however is already short staffed and has to frequently ask for more information than the initial engineering request contains. As a result, engineering requests can take days to be completed. The LLM system can help in creating better engineering requests and SAP entries. additionally, an LLM system can also search through available sources more efficiently and with context instead of exact word matches. This holds the opportunity to reduce the time of repair significantly, although this effect still has to be proven operationally.

SAP data quality: While initial expectations for the integration of SAP data, particularly historical maintenance records, were optimistic, its actual contribution proved limited. Although early user interviews prompted some adjustment of these expectations, the SAP extract was rarely retrieved during application use. Most queries posed by users could be answered using technical manuals or remained unresolved.

The limited value of SAP data is assumed to be a consequence of the variable quality and completeness of the SAP records, which, being in Dutch, might also have influenced embedding and retrieval. As a result, the incorporation of historical maintenance data contributed little to the system's overall performance.

After all, generating quality answers necessitates quality data. Documentation quality of maintenance work varies between different crews onboard of different ships, depending on enforcement of working principles. It was discovered that crews see a lot of potential in not only retrieving information from the system, but also using the system to generate detailed descriptions of the work performed. These descriptions can be put back in SAP through a supplied 'copy-paste' function. This saves time, and more importantly, increases the quality of data that is contained in the SAP system.

Context is vital: the manuals contain a lot of irrelevant information, for example about different configurations of propulsion systems. We found that it helps immensely to put context of the operational environment in the system, indicating the configuration in use, the ship that the operator is on (relevant for maintenance data) and the role and experience of the operator. This makes it more easy to ask the system questions and get relevant answers, while removing the need to provide endless details.

A LLM is text oriented, numbers are more difficult. Numbers require another approach to reduce hallucinations. This poses problems if users ask about values such as valve pressure or oil temperatures. This also poses challenges to include real-time operational data from the propulsion system into the system, albeit not impossible.

From diagnosis, to recovery, to finalization: From the design thinking interviews and the analysis of shipboard business operations, we identified three phases in which a LLM system can assist. In case of a calamity, the first step is to diagnose the problem. The system proposes ways to do so based on symptoms that are entered by an operator. Secondly, after identification of the problem, the system proposes a action list to be accepted by the operator and indicates what components are needed for maintenance work. During recovery, the user can ask additional questions and use tick-boxes to indicate the process. Finally, after the system is fully recovered, the system helps to finalize the work by providing a work summary and SAP entries.

The design thinking methodology led to the following design, focusing on the conversation, history, feedback, and the three phases and support during the recovery process.

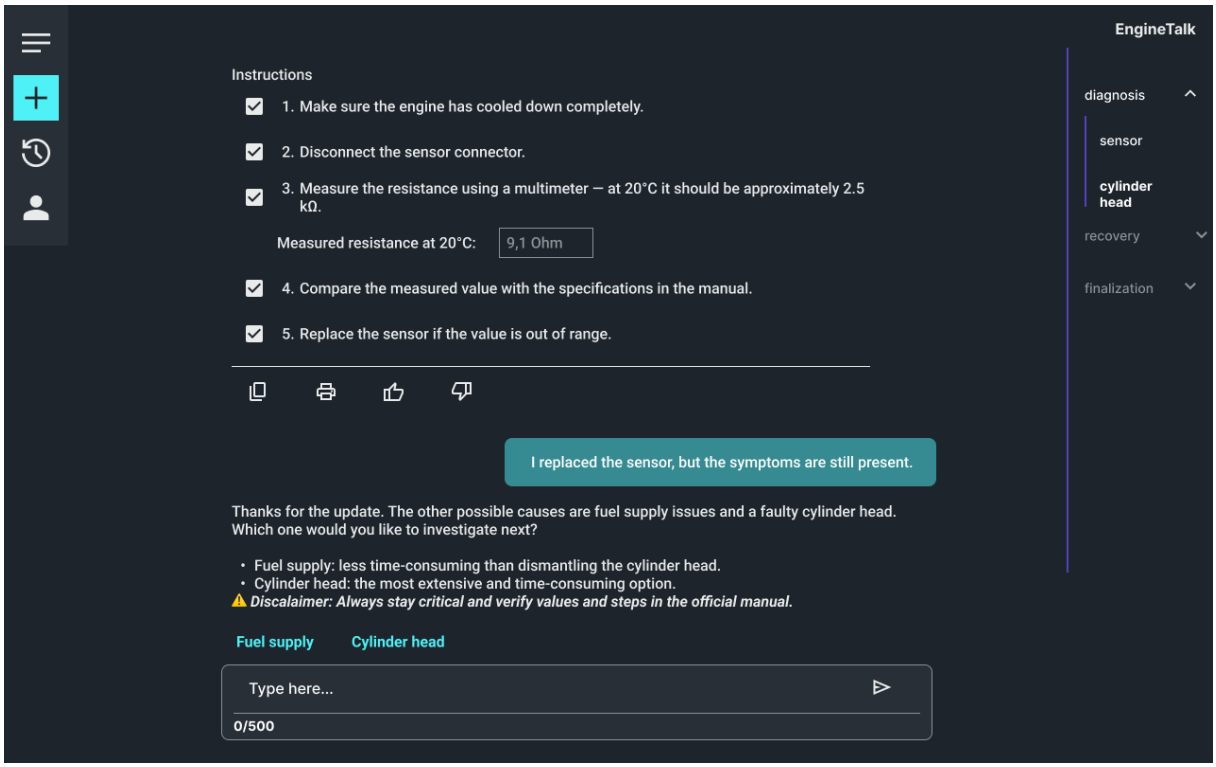

Figure 1: Design of the EngineTalk interface. In the middle, a chat window is visible in recovery mode

Overall, the end-users were enthusiastic[4] to give input and test EngineTalk, and were eager to see our solution deployed as part of their available utilities. There was an overall consensus that relevant information could be retrieved much quicker, which saves a lot of time. Moreover, they suggested that communication between the ship and remote support would be less necessary, reducing workload and decreasing repair times.

3.2. Technical results

Indeed, the employment of off-line, local LLM capacities introduces complexities and constraints which prove challenging, albeit not impossible, to overcome.

For example, as EngineTalk relies considerably on query-reformulation techniques and test-time translation, there was a considerable overhead in the amount of LLM response-generations, and multiple retrieval searches. Moreover, the LLM system was subject to a context window limit of 8.000 tokens. Though this is sufficient for any single interaction, any multitude of consecutive dialogue, each which may also retrieve context information, may exceed such limits for long conversations that demand retrieval of context chunks of significant lengths.
Nevertheless, despite such challenges EngineTalk proved effective in utilizing local LLM models with RAG to meet the needs of on-board engineers, and answer various queries spanning the domain of both technical manuals as well as historical SAP maintenance records. Though hallucinations inevitably do occur, these were not observed to be present to any sufficiently prevalent degree that renders EngineTalk defunct. End-user validation indicated that our system outputs were in-line with both their expectations and demands, especially when also providing a direct link to source documentation.

3.3. Roadmap results

Future developments focus on several main points, identified from the user feedback sessions:

[4] A notable quote (paraphrased from Dutch) of a crew member during a test was *"I'm happy to participate. Actually, I'm on leave this week but I liked to make time to participate in this test because I'm so enthusiastic about this development!"*

Deploying EngineTalk on ship hardware: Currently the system is deployed as a remotely-available application, which is hosted on-shore with capable hardware and accessible through a secure connection from naval vessels. However, such off-board utilization does not meet the demands for operations in disconnected environments. Therefore, we plan to deploy EngineTalk within a containerized environment with a dynamic interface to shared LLM resources onboard of the ship. This design infrastructure enables various LLM-driven applications to utilize the same LLM-supporting hardware.

Creating LLM systems for different usergroups aboard the ships: the system has in principle proven useful for part of the technical crew of a navy ship, aimed at engine maintenance. Users see potential use-cases for other user groups that are also performing maintenance at other systems within the ship, such as maintenance on electrical distribution.

Given the limited retrieval utility of historical SAP maintenance records, future work should focus on improving the quality and consistency of these records at the source (by leveraging an LLM to fill the records). Exploring more advanced embedding strategies or fine-tuning domain-specific language models might improve the system's ability to extract relevant context from less structured or variable-quality data. Finally, user interaction data could inform the development of more targeted indexing strategies, prioritizing content types that have shown higher utility in practice.

Integrating EngineTalk with other operational IT systems: due to the reliance on SAP data for giving insight in maintenance data, we also aim to explore the potential deeper integration of SAP into the system, ideally by enabling the LLM to use SAP as a tool. Using that interface, EngineTalk can directly browse and consult up-to-date maintenance data, and the system can more easily be used for creating quality maintenance records to be put back in SAP. Moreover, we are similarly eager to integrate graph-based RAG technologies (known as GraphRAG, as described in (Peng *et al.* 2024)), which can both enhance conventional RAG by enabling a pre-processing or pre-filtering of chunk retrieval, as well enable integration with various other graph-compatible data sources.

Using LLMs to interpret real-time data sources: a lot of data is collected in real-time from sensors in the ship. These sensor outputs are used for analysing the performance of the propulsion systems and can be used to generate alarms, diagnose issues, or indicate potential future anomalies. For an LLM it is difficult to reason about real-time data, but combined with other software concepts such as linked data, they have the promise to translate data into valuable insights.

Mitigating hallucinations: while the utilization of RAG significantly reduces the risk of hallucinations, this risk is not fully mitigated entirely, despite the LLM granted access to grounded-context material. Combined with the risk of automation bias among crew members, we were compelled to put greater emphasis on such output risks, and stimulate end-users to consult the relevant retrieved material directly, as to validate the model's outputs by a human-in-the-loop. However, in order to further reduce hallucinations, future development will explore additional hallucination-mitigating techniques such as chain-of-thought verification, uncertainty estimation, and post-processing output corrections (previous work from (Tonmoy *et al.* 2024) explores a considerable collection of advanced hallucination mitigations).

4. Conclusion

This paper investigates the development and implementation of EngineTalk: a Retrieval Augmented Generation (RAG) enabled LLM designed to support engine maintenance crews on RNLN vessels. The findings demonstrate that EngineTalk proves the added value of using LLMs for maintenance aboard of ships to be promising. Initial results from user feedback sessions aim in this direction, with the main benefit being that a lot of information can now be easily accessible. This holds value in increasing efficiency of maintenance work, and in decreasing the level of expertise needed to start on maintenance work.

A combined approach of integration of both maintenance documentation (manuals) and maintenance data (SAP records) helped in gaining these results, and further integration of SAP and EngineTalk is recommended to generate responses based on live-data. Also, such an integration would make it possible to feed quality maintenance data back into SAP and make that process more pleasurable, as it is often noted that the current process is a hassle and for the maintenance crew and has a risk of being neglected.

From a technical perspective the specialised approach of EngineTalk proved successful as well, as in just 6 months time a Minimal Viable Product could be created with a decent accuracy and precision in generating results to user questions. The combination of RAG and LLMs helped to reduce errors and hallucinations, although it should be noted that users are still advised to always consult the linked source material as well. The loosely coupled

architecture between the LLM, the RAG (and the vectorized data) and the frontend creates a situation in which the system can be quickly re-used in a different environment with new user groups, by only changing one of the components. Nevertheless, regarding deployment of models would be to consider commercially available components unless a highly specialised tool is needed due to the time and effort required to tailor open source models to make them fit for use.

Considering the increasing complexity of naval systems and the persistent shortage of technically skilled personnel, EngineTalk demonstrates the potential of LLM-based support systems to enhance operational effectiveness. EngineTalk represents a practical foundation for the broader integration of AI-driven tools in maritime maintenance, supporting both workforce resilience and mission readiness in a rapidly evolving operational environment.

Acknowledgements

The authors would like to thank Aletta van Straaten, Kirty Bol, Maarten Kuppers, Nadine Haagman, Nathan Neelis, Rick van Hattum, Rick Verweij, Saranda Kienjet whose contributions were invaluable in the development of this work.

References

Brown, T., 2009. *Change by Design: How Design Thinking Creates New Alternatives for Business and Society.* Harvard Business Press.

Es, S., James, J., Espinosa-Anke, L., and Schockaert, S., 2025. Ragas: Automated Evaluation of Retrieval Augmented Generation.

Graham, Y., 2015. Re-evaluating Automatic Summarization with BLEU and 192 Shades of ROUGE. *In*: L. Màrquez, C. Callison-Burch, and J. Su, eds. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Presented at the EMNLP 2015, Lisbon, Portugal: Association for Computational Linguistics, 128–137.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W., Rocktäschel, T., Riedel, S., and Kiela, D., 2021. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks.

Liedtka, J., 2015. Perspective: Linking Design Thinking with Innovation Outcomes through Cognitive Bias Reduction. *Journal of Product Innovation Management*, 32 (6), 925–938.

Peng, B., Zhu, Y., Liu, Y., Bo, X., Shi, H., Hong, C., Zhang, Y., and Tang, S., 2024. Graph Retrieval-Augmented Generation: A Survey.

Reiter, E., 2018. A Structured Review of the Validity of BLEU. *Computational Linguistics*, 44 (3), 393–401.

Steck, H., Ekanadham, C., and Kallus, N., 2024. Is Cosine-Similarity of Embeddings Really About Similarity? *In*: *Companion Proceedings of the ACM Web Conference 2024*. 887–890.

Tiddens, W., Ten Zeldam, S., Curvers, D., Zegers, J., and Pollmann, B., 2024. Towards a data-driven naval maintenance organisation: the importance of a social roadmap. *In*: *Conference Proceedings of INEC*.

Tonmoy, S.M.T.I., Zaman, S.M.M., Jain, V., Rani, A., Rawte, V., Chadha, A., and Das, A., 2024. A Comprehensive Survey of Hallucination Mitigation Techniques in Large Language Models.

Voûte, R., Smit, B.P., and Linden, Y., 2024. Improving the internal battle in a navy ship by adding situation awareness by means of using a 3D geospatial model combined with a linked data model of this ship. Design phase. *Conference Proceedings of INEC*.

van Zijl, L., Vermeulen, J., and Linden, Y., 2024. Enhancing Internal Battle Operations Through the Battle Damage Repair Tool. *Conference Proceedings of INEC*.