18th International Conference on Space Operations, Montreal, Canada, 26 - 30 May 2025.
Copyright 2025 by EUMETSAT. Published by the Canadian Space Agency (CSA) on behalf of SpaceOps, with permission and released to the CSA to publish in all forms.

SpaceOps-2025, ID # 413

Improving Systems Engineering through AI Assistance

Özgür Ünalan[a]*, Alexander Rudolph[b], Oriol Hinojo Comellas[c]

[a] *EUMETSAT, Eumetsat Allee 1, Darmstadt, Germany*, *oezguer.uenalan@eumetsat.int*
[b] *EUMETSAT, Eumetsat Allee 1, Darmstadt, Germany*, *alexander.rudolph@eumetsat.int*
[c] *EUMETSAT, Eumetsat Allee 1, Darmstadt, Germany*, *oriol.hinojocomellas@eumetsat.int*

* Corresponding Author

Abstract

This paper introduces an innovative AI-assisted tool designed to address significant challenges in Systems Engineering (SE), particularly the time-consuming nature of verification tasks and inconsistent quality of SE artifacts due to resource constraints. The authors present SysAssist, a comprehensive web-based solution that integrates Large Language Models (LLMs) to support Systems Engineers throughout the SE lifecycle. Developed at EUMETSAT, the tool employs a user-friendly interface enabling engineers to leverage AI capabilities without requiring specialized knowledge of prompt engineering or AI workflows. The system provides three primary functionalities: requirements review based on INCOSE criteria, design adequacy assessment, and requirements allocation and flow-down analysis. SysAssist has been successfully implemented in several projects, including the System Requirements Review, Critical Design Review, and an operational planning tool requirements assessment, demonstrating potential time savings of 10-30% in verification and validation tasks. The development philosophy emphasizes supporting engineers rather than replacing them, with AI outputs requiring human verification to address potential limitations in AI reliability. The dual-environment architecture balances performance with data protection by utilizing on-premises processing for sensitive information and cloud-based services for enhanced capabilities with publicly available data. Prompt engineering plays a critical role in the system's implementation, with specialized templates developed for different analysis types to maximize accuracy and relevance of AI responses. Organizational feedback indicates that providing the tool to requirement authors rather than reviewers prevents duplication of findings and improves quality early in the development process. Future developments include expanding capabilities through Retrieval-Augmented Generation (RAG), implementing agent-based approaches for complex workflows, and integrating with existing SE tools. The authors conclude that SysAssist represents an early step toward a hybrid approach where AI and human engineers collaborate effectively, enhancing rather than replacing human expertise in the complex domain of Systems Engineering.

Keywords: Systems Engineering, Artificial Intelligence, Requirements Verification, Process Automation, Large Language Models, Quality Assurance

Nomenclature

This section is not numbered. A nomenclature section could be provided when there are mathematical symbols in your paper. Superscripts and subscripts must be listed separately. Nomenclature definitions should not appear again in the text.

Acronyms/Abbreviations

Abbreviation	Definition
AI	Artificial Intelligence
API	Application Programming Interface
CO2M	Carbon Dioxide Monitoring mission
EPS	EUMETSAT Polar System
EUMETSAT	European Organisation for the Exploitation of Meteorological Satellites
INCOSE	International Council on Systems Engineering
LLM	Large Language Model
LoRA	Low Rank Adaptation
MCC	Mission Control Center

Copyright 2025 by EUMETSAT. Published by the Canadian Space Agency (CSA) on behalf of SpaceOps, with permission and released to the CSA to publish in all forms.

MCOS	Mission Control and Operations System
OPS	Operational Planning System
RAG	Retrieval Augmented Generation
REQ-VV	Requirements Verification and Validation
RVE	Requirements & Verification Engineering
SE	Systems Engineering
SysML	System Modelling Language
TEN	Technical Note

1. Introduction

As the intergovernmental agency responsible for monitoring weather and climate from space, the European Organisation for the Exploitation of Meteorological Satellites (EUMETSAT) manages a fleet of 10 satellites spanning diverse programmes, with additional missions planned for launch in the coming years.

In this complex operational environment, Systems Engineering (SE) faces challenges as systems become increasingly complex. While well-established SE processes and knowledge exist, executing them thoroughly requires substantial time and resources. Stakeholders simultaneously expect greater agility, facilitated reuse, and architectures supporting multipurpose missions.

The fundamental problem lies in time constraints that force engineers to conduct selective or sample-based reviews, often resulting in quality issues and incomplete analyses. This paper introduces an innovative AI-assisted solution called "SysAssist" that originated from supporting Requirements and Verification Engineering at EUMETSAT. SysAssist, as the name suggests, serves as a comprehensive AI-powered assistant for Systems Engineering tasks, providing automated support for various verification and validation activities.

Initially focused on reviewing requirement quality based on INCOSE guidelines, the tool quickly expanded to additional use cases such as design review validation against requirements. It became evident that a highly technical solution would benefit only those with specialized knowledge. Therefore, we developed a user-friendly platform enabling Systems Engineers to effectively leverage AI technology without requiring in-depth expertise in prompt engineering or other AI techniques.

This paper aims to share our vision for an AI-based Systems Engineering assistant, document the challenges encountered, outline future directions, and describe our plan for organizational implementation through a professional, service-based approach. We believe many organizations are implementing similar solutions and welcome knowledge exchange with the broader systems engineering community.

2. Background and Approach

2.1 Development Philosophy: Supporting Engineers Without Replacing Them

Systems Engineering is a well-established discipline with numerous organizations contributing to its knowledge base. INCOSE brings together engineers worldwide to collaboratively develop systems engineering processes and standards across the entire lifecycle. Currently, the Systems Engineering community is actively exploring effective applications of artificial intelligence within the domain.

As this exploration is in its early stages, the optimal uses of AI for Systems Engineering activities remain somewhat undefined. However, AI's limitations—such as hallucinations, context length constraints, and knowledge cutoff—must be addressed, especially in a critical discipline where correctness and reliability are paramount.

Additionally, as systems become more complex and engineering teams face expanding workloads, AI offers an opportunity to scale engineers' capacity. Rather than replacing engineers, AI tools can transform their roles, allowing them to oversee a wider scope of activities while delegating routine verification tasks to AI assistants. This shift in responsibilities enables engineers to focus on high-value judgment activities that require human expertise while still maintaining oversight of AI-assisted processes.

2.2 Current State of Systems Engineering Processes

The development philosophy of SysAssist is to support engineers in labour-intensive work rather than replace them. This approach originates from several considerations. AI (as human intelligence also) is not always 100% reliable regarding hallucinations or fabricating information not originally provided. Our current philosophy relies on AI to review existing Systems Engineering artifacts (primarily requirements) with engineers cross-checking the AI's output.

Copyright 2025 by EUMETSAT. Published by the Canadian Space Agency (CSA) on behalf of SpaceOps, with permission and released to the CSA to publish in all forms.

Although AI can process and review many artifacts quickly, producing substantial information requiring human verification, we believe this verification effort is less intensive than conducting reviews entirely without AI support. We anticipate that newer AI models will progressively reduce hallucination issues over time.

In our current implementation at EUMETSAT, our Requirements & Verification Engineering team has applied SysAssist to support 5-6 substantial requirements reviews. Each review typically involved assessing approximately 400 requirements against 5 quality criteria, demonstrating the tool's ability to handle significant workloads while remaining under expert supervision to ensure output quality. This supervised approach ensures quality while we work toward our target of 95% correctness, after which we plan to provide the tool directly to requirement authors. Even in this supervised mode, we have observed that early AI-assisted reviews help identify issues before formal reviews, improving the overall efficiency of the review process.

We believe these tools should be provided to the authors or creators of requirements rather than reviewers (for example, those reviewing a requirements document in a system requirements review [4]). Otherwise, multiple reviewers using the same AI tool might raise duplicate findings. More value is added when authors can identify weaknesses in their requirement specifications and designs before formal expert-based reviews, thereby raising overall quality.

Our aim is to support the entire Systems Engineering lifecycle process, covering not only requirements but also design and verification. We believe implementation aspects are well-addressed by existing technologies for coding and data analysis.

2.3 User-Centred Design Approach and implementation strategy

We've designed SysAssist with a user-friendly interface and web application architecture to maximize accessibility. Currently, leveraging AI effectively requires users to provide the right context and prompts in the appropriate sequence to achieve desired results. Our approach abstracts this complexity, allowing Systems Engineers to focus on their primary work rather than AI mechanics.

Our implementation strategy is iterative, beginning with a prototype that we demonstrate to potential users for valuable feedback, followed by improvements to existing use cases. Over time, we plan to add more use cases along the Systems Engineering lifecycle to provide end-to-end AI support for Systems Engineering processes.

2. System Architecture and Implementation

3.1 Overall Architecture of the AI-Assisted Tool

SysAssist employs a dual-environment architecture designed to balance performance needs with data protection requirements. The following diagram shows an overview of the current architecture:

Copyright 2025 by EUMETSAT. Published by the Canadian Space Agency (CSA) on behalf of SpaceOps, with permission and released to the CSA to publish in all forms.

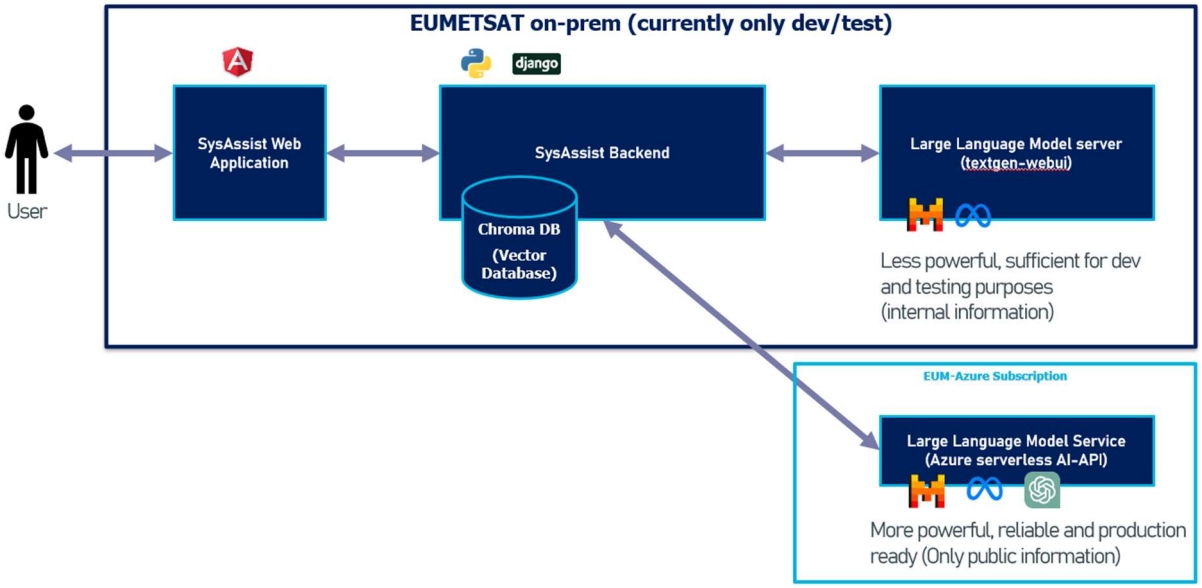


Fig. 1. Architecture of SysAssist

The user interacts with the Web application that provides several use cases to the users (see subsections of section 4). The system consists of two main deployment options:

1. **EUMETSAT On-Premises Environment (Development/Testing)**
   - SysAssist Web Application (Angular frontend): Provides an interactive user interface for querying and interacting with the AI system.
   - SysAssist Backend (Django/Python): Manages user requests, orchestrates AI processing, and integrates system components.
   - Vector Database (Chroma DB): Enables efficient context retrieval by storing and retrieving embeddings of internal documents.
   - Large Language Model (text-generation-webui [9]): Hosts an on-premises LLM for generating AI responses while ensuring internal data privacy.
   - Optimized for internal use: This setup prioritizes handling confidential information while operating with limited computational power.

2. **EUM-Azure Subscription Environment**
   - Large Language Model Service (Azure serverless AI-API): Provides a scalable, high-performance LLM service.
   - Optimized for reliability and production use: This setup is designed for being more powerful, reliable, and production-ready while ensuring security compliance only being use on processing public information.

The two environments can be used alternatively, with the backend communicating with either the on-premises LLM or the cloud-based service depending on data sensitivity and computational requirements. This architecture follows a client-server model with a web-based frontend allowing users to interact with the system through various specialized workflows designed for specific Systems Engineering tasks. This hybrid approach ensures both efficiency and security.

3.2 Web Application Implementation

The solution is implemented as a web application with a Python backend using the Django framework. This choice was driven by Python's extensive ecosystem for AI and machine learning, facilitating integration with various

Copyright 2025 by EUMETSAT. Published by the Canadian Space Agency (CSA) on behalf of SpaceOps, with permission and released to the CSA to publish in all forms.

AI components. The frontend utilizes the Angular web application framework to provide a responsive and interactive user experience.

The application allows users to upload file based exports of Systems Engineering artifacts (Excel spreadsheets, Word documents) for subsequent analysis by AI, significantly reducing manual effort for routine verification tasks.

Our implementation leverages Large Language Models (LLMs) with different system prompts tailored to specific models and use cases. The overall prompt architecture is composed of distinct components:

- System prompt defining baseline behaviour – Example: "You are an expert systems engineer providing helpful assistance"
- Role/persona assignment clarifying the AI's perspective – Example: "Act as a worldclass requirements engineering expert"
- Specific task descriptions outlining the exact analysis required – Example: "Critically check if the system design description in the provided context adequately addresses a design for all aspects of the following requirement"
- Context provision (requirements documents, specific requirements, design specifications, etc.)
- Placeholders for dynamic content insertion

Prompt engineering plays a critical role in maximizing the accuracy and relevance of AI responses, requiring careful tuning for each specific use case. We've created specialized prompts for different analysis types using also prompt techniques such as the CO-STAR prompting technique [1].

3.2 Data Protection and Confidentiality Measures

To address data confidentiality concerns, we've implemented self-hosted LLM solutions using frameworks like Ollama [8] and Text-generation-webui [9]. We maintain strict controls on document processing, avoiding the transmission of confidential documents to external services. Our dual-environment approach ensures that sensitive internal data remains on-premises, while the more powerful cloud-based processing is used only for non-sensitive information

3.3 Integration Potential with Existing SE Tools

While not yet fully implemented, our architecture envisions API endpoints for integration with existing Requirements Management tools such as DOORS. This would allow engineers to receive immediate feedback within their regular workflows without switching contexts. Future plans include development of specialized agents for various systems engineering tools, including:

- M-Files agent for document retrieval
- DOORS agent for requirements management
- Enterprise Architect agent for modelling

The integration will initially focus on file-based exchanges (Excel, Word) with a future transition to direct API integration as the system matures.

4. Core Features and Functionality

Before diving into the specific use cases, the areas in which the AI assistance is currently implemented is depicted in the following overview (noted with "AI") in the context of the Systems Engineering Lifecycle:

Copyright 2025 by EUMETSAT. Published by the Canadian Space Agency (CSA) on behalf of SpaceOps, with permission and released to the CSA to publish in all forms.

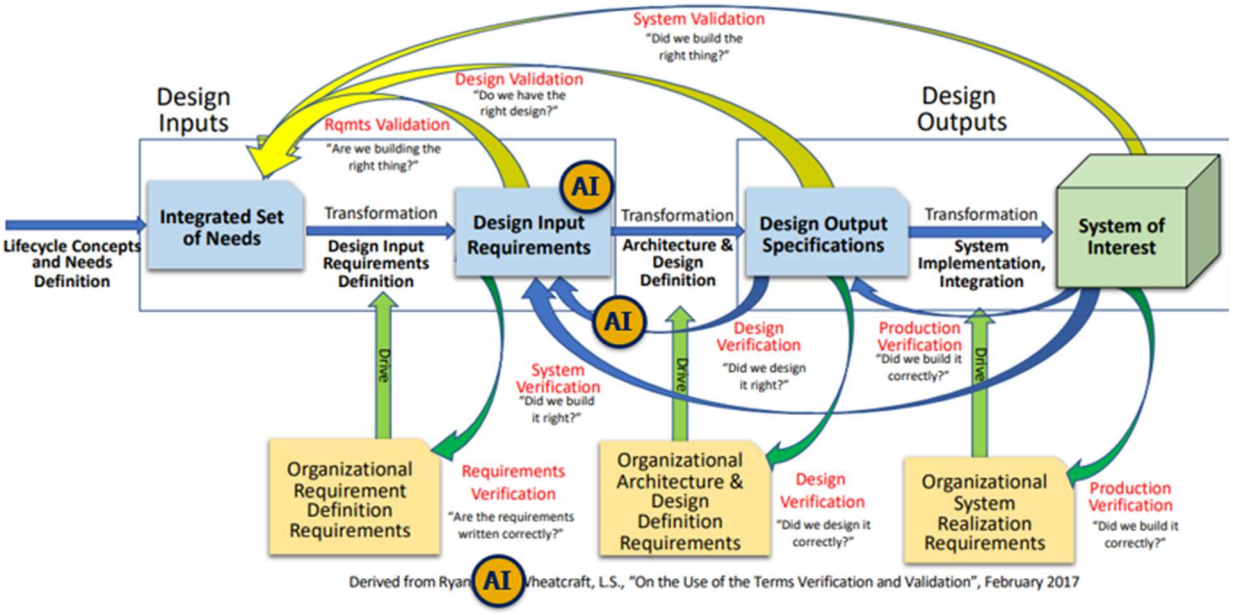


Fig. 2. AI assistance in the System Engineering Lifecycle [2]

The use cases implemented can be mapped to the following questions in the System Lifecycle:

- "Are the requirements written correctly?" (see 4.1 and 4.3)
- "Did we design it right?" (see 4.2)

The following screenshot shows the start page of the SysAssist application and showing the different use cases, it provides:

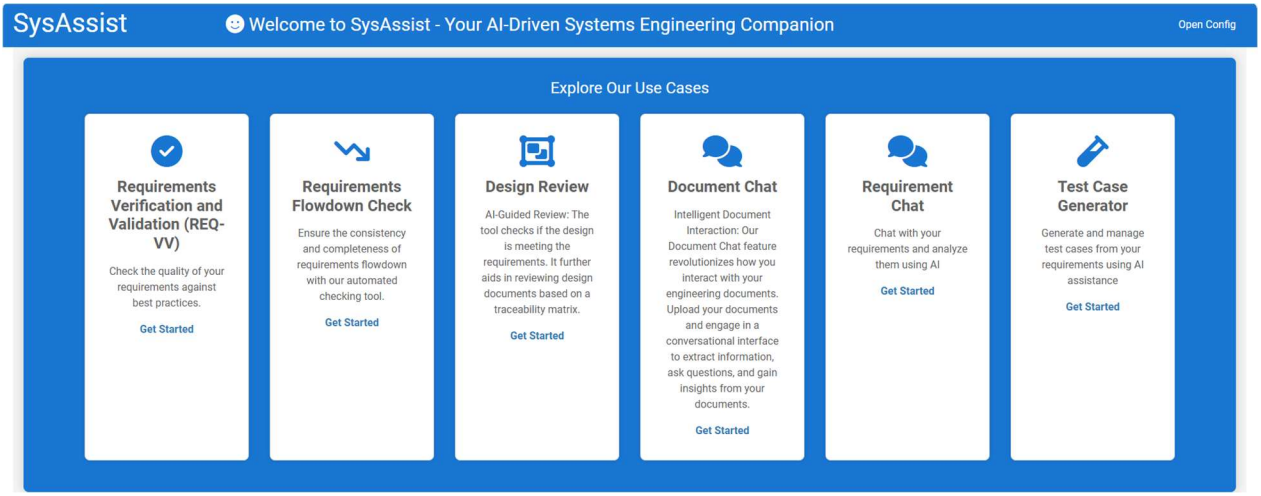


Fig. 3. AI assistance in the System Engineering Lifecycle

In the following sections, only the most mature use cases will be presented that already produced some evaluation results and that are used in practice.

4.1 Use Case1: Requirements Review Using EUMETSAT/ INCOSE Criteria

Copyright 2025 by EUMETSAT. Published by the Canadian Space Agency (CSA) on behalf of SpaceOps, with permission and released to the CSA to publish in all forms.

The requirements review functionality evaluates input requirements based on established INCOSE criteria [2] for well-formed requirements. The tool provides:

- Live preview capabilities allowing users to see results immediately but also enables downloading whole reports on requirements set.
- Specific issues identification regarding clarity, testability, and completeness.
- Concrete suggestions for requirement improvement.
- References to specific INCOSE guidelines supporting each recommendation.

Each requirement is systematically analysed against six requirements criteria (RC)including singularity, style and clarity, unambiguity, completeness, verifiability, and correctness, with specific improvement suggestions provided. The system highlights both conforming and non-conforming aspects of each requirement and provides revised requirement text when issues are identified. The criteria can be easily expanded if needed. Following is a screenshot of the application.

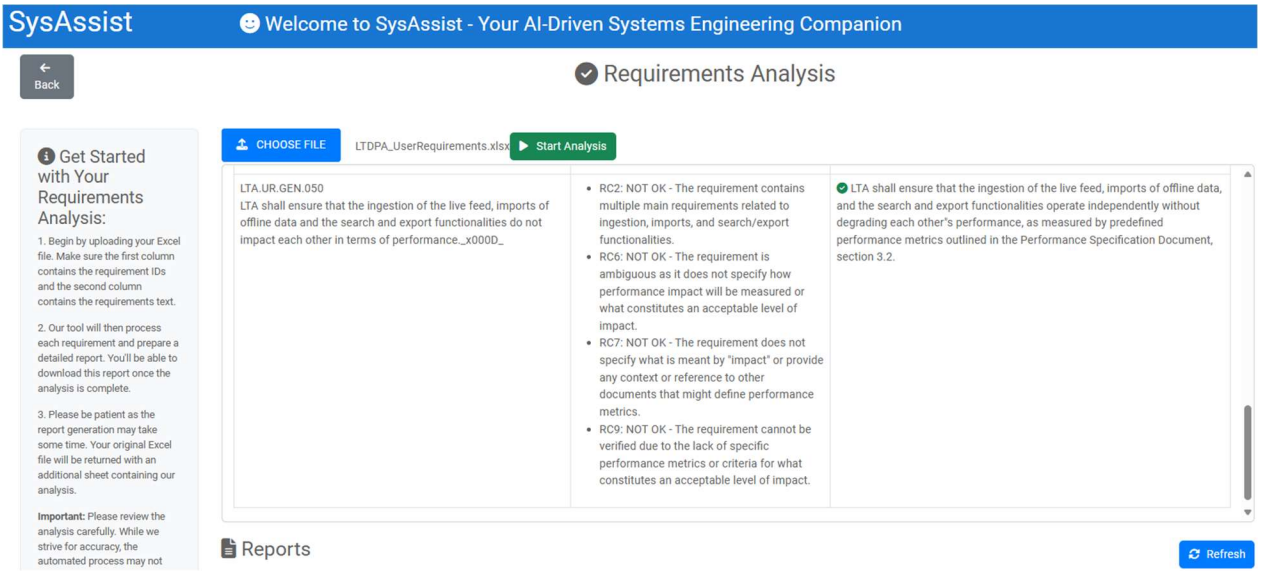


Fig. 4. Screenshot of the Requirements review feature

The following criteria will be checked:

- **RC2** - Requirements quality criteria: **Is singular**
  - The requirement statement **addresses a single capability, characteristic, constraint, or quality factor**. This can be evaluated by answering the question:
    - Does the statement/sentence **only state exactly one requirement** and not multiple requirements?
- **RC3** - Requirements quality criteria: **Is conforming**
  - The requirement **conforms to an approved standard pattern** and **style guide or standard** and can be answered by means of the following questions:
    - Is **active voice** used?
    - Is the requirement stated in the **same style as** the **other** requirements?
    - Are the requirements written by using a **uniform language**?
- **RC6** - Requirements quality criteria: **Is unambiguous**
  - Requirement statements is stated such that the **requirement can be interpreted in only one way**. Questions to assess this are:
    - Are **multiple interpretations** of the requirement **possible**?
    - Is the **rationale** for the requirement available?

Copyright 2025 by EUMETSAT. Published by the Canadian Space Agency (CSA) on behalf of SpaceOps, with permission and released to the CSA to publish in all forms.

        - Are there are any "**weak words**" in the requirement that can **indicate ambiguity**, for example: "and", "or", "if" without "else" or any other so called "weak word"
- **RC7** - Requirements quality criteria: **Is complete**
    - The **requirement statement sufficiently describes** the necessary capability, characteristic, constraint, or quality factor to meet the need **without needing other information to understand the requirement.** Questions to assess this are:
        - Is the requirement **understandable on its own?**
        - Are there **referrals** to other requirements or other documents?
        - Are **referred documents specific to sections** and not referring the document as a whole?
        - Are there any **TBCs (to be confirmed), TBDs (To be done)** or generally TBx existing?
- **RC9** - Requirements quality criteria: **Is verifiable**
    - The requirement is **structured and worded such that its realization can be proven (verified)** at the level the requirement exists. The following can be asked to check this:
        - Does the link to a **verification case** or **a verification attribute exists**?
        - Can the requirement be verified?
        - Are clear acceptance criteria definable for the requirement?
        - Is the requirement written to meet concrete values or value ranges (i.e. timeliness of data dissemination in minutes)

In order to check all of this criterion a prompt with parameters ({text}, {system_name}*)* was generated to check:

"Evaluate the given requirement based on the following criteria, addressing each criterion separately. Your response should be concise, clear, and focused on identifying any issues related to the criteria.

RC2 - Singularity: Assess if the requirement contains only one main requirement. Check for the use of phrases like 'and', 'or', etc., which might indicate multiple requirements.
RC3 - Style and Clarity: Determine if the requirement is written in the style of 'The system shall', using active voice. Ensure the system responsible for implementing the requirement is clearly mentioned.
RC6 - Unambiguity: Evaluate if the requirement could be interpreted in multiple ways.
RC7 - Completeness: Consider whether the requirement is understandable on its own. Identify if there are any referrals to other requirements or documents and whether they specify particular sections or pages. Look for any TBCs, TBDs, TBRs, or general TBx.
RC9 - Verifiability: Decide if the requirement can be verified, particularly considering the specificity of any document references.

Present your findings for each criterion succinctly with a verdict: "OK" or "NOT OK". If a criterion is fully met, do not output anything! Otherwise, list the specific issues.

Requirement for Evaluation: '{requirement}'. Please provide a structured evaluation with your findings and recommendations (maximum one sentence for each criteria). Provide a improved requirement sentence to address the findings, but only if the requirement is not OK. The system name is {system_name}."

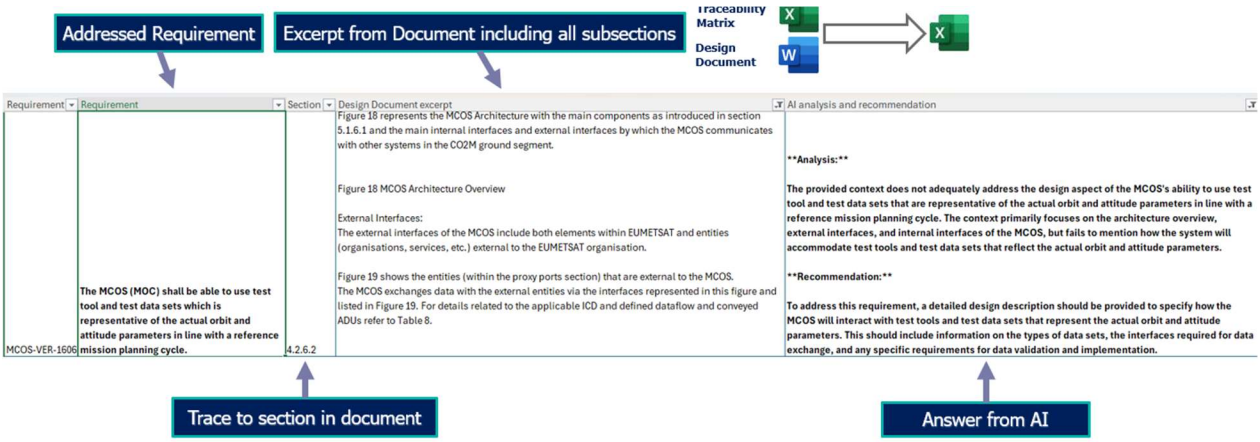
4.2 Use Case 2: Design adequacy assessment

The design review feature analyses whether design descriptions adequately address their corresponding requirements. The system works by:

1. Taking in design documents (Word format) and traceability matrices (Excel) that map the design section to requirements that they address
2. Using optimized prompts to evaluate whether design elements properly address their corresponding requirements
3. Analysing the context of design descriptions in relation to the requirements they should fulfil. For this purpose, the text within a section of a design description and all subsections are provided to check against the requirement the design should fulfil.
4. Providing detailed analysis with clear recommendations when design elements are incomplete

Copyright 2025 by EUMETSAT. Published by the Canadian Space Agency (CSA) on behalf of SpaceOps, with permission and released to the CSA to publish in all forms.

The following figure shows an example of the analysis:



Requirement	Requirement	Section	Design Document excerpt	AI analysis and recommendation
MCOS-VER-1606	The MCOS (MOC) shall be able to use test tool and test data sets which is representative of the actual orbit and attitude parameters in line with a reference mission planning cycle.	4.2.6.2	Figure 18 represents the MCOS Architecture with the main components as introduced in section 5.1.6.1 and the main internal interfaces and external interfaces by which the MCOS communicates with other systems in the CO2M ground segment. Figure 18 MCOS Architecture Overview External Interfaces: The external interfaces of the MCOS include both elements within EUMETSAT and entities (organisations, services, etc.) external to the EUMETSAT organisation. Figure 19 shows the entities (within the proxy ports section) that are external to the MCOS. The MCOS exchanges data with the external entities via the interfaces represented in this figure and listed in Figure 19. For details related to the applicable ICD and defined dataflow and conveyed ADUs refer to Table 8.	**Analysis:** The provided context does not adequately address the design aspect of the MCOS's ability to use test tool and test data sets that are representative of the actual orbit and attitude parameters in line with a reference mission planning cycle. The context primarily focuses on the architecture overview, external interfaces, and internal interfaces of the MCOS, but fails to mention how the system will accommodate test tools and test data sets that reflect the actual orbit and attitude parameters. **Recommendation:** To address this requirement, a detailed design description should be provided to specify how the MCOS will interact with test tools and test data sets that represent the actual orbit and attitude parameters. This should include information on the types of data sets, the interfaces required for data exchange, and any specific requirements for data validation and implementation.

Fig. 5. Excerpt of a design check against the requirements it should fulfil

For example, as shown above, when analysing a requirement for MCOS components, the system identified that the design description lacked clear explanation of how test tools and test data item will be handled based on the design description and recommended specific additions to remedy this gap. The approach contributes to ensure that all requirements have proper design coverage before proceeding to implementation.

4.3 Use Case 3: Allocation and Flow-Down Analysis

This feature checks the allocation and flow-down of requirement by checking if lower-level requirements properly address higher-level ones. The analysis includes:

- Traceability verification between requirement levels
- Completeness assessment of requirement decomposition
- Identification of orphaned or inadequately covered requirements
- Verification that derived requirements maintain consistency with parent requirements

The system uses a structured approach like the design review functionality, checking lower-level requirements against their parent requirements, identifying specific issues like incomplete coverage or inconsistencies, and providing explicit recommendations for improving requirement decomposition. Here is an example of the output, where a child requirement is checked whether it fully addresses the parent requirement:

Parent Requirement	Child Requirements	LLM Result
		Analysis: The child requirement '[MDPSRD] MDPS-FUN-PDP-MPI-10320' focuses on maintaining production completeness during the addition or modification of product generation functions. While this is a critical aspect, it does not fully address the additional requirement for maintaining timeliness of the product generation process during routine maintenance and upgrade activities, an aspect emphasized in the parent requirement '[CO2M OGSRD] CO2M-OGS-REQ-0094'.
[CO2M OGSRD] CO2M-OGS-REQ-0094 In Nominal Processing mode, the Process Instrument data and generate CO2M products function shall allow performing routine maintenance and upgrade activities without affecting the timeliness and production completeness.	[MDPSRD] MDPS-FUN-PDP-MPI-10320 It shall be possible to add new L0/L1/L2 Products Generation functions or change the configuration of the existing ones without affecting the completeness of the production.	Judgement: NOK. Recommendation: The child requirement should be updated to ensure that any routine maintenance and upgrade activities, including adding new product generation functions or

Fig. 6. Screenshot of Allocation and Flow-Down Analysis result

Copyright 2025 by EUMETSAT. Published by the Canadian Space Agency (CSA) on behalf of SpaceOps, with permission and released to the CSA to publish in all forms.

Please note that there can be multiple subsystems and subsystem requirements addressing the parent requirements. This is quite challenging, also for an AI, to assess whether all aspects of the parent requirements are addressed in the child requirements, but the different aspects of the parent requirement might be distributed to several requirements.

The user can check the recommendation of the SysAssist to enhance the correctness of the allocation and flow-down of parent requirements to child requirements.

4.4 Mechanisms for Highlighting Critical Findings

While not yet fully implemented, our planned approach for highlighting critical findings includes:

- A classification system categorizing issues as critical, major, or minor based on their potential impact. This depends on the use case and must be incorporated into the existing prompt or prompt chain to have the AI also classify the issues
- Leveraging user feedback through upvote/downvote functionality to refine criticality assessments. This can be later used to fine-tune the prompt or also the model itself with techniques like LoRA (Low-Rank Adaptation) [7].
- Visual indicators in the interface that highlight high-priority findings requiring immediate attention

This feature will address the challenge of managing the potentially large number of findings generated by comprehensive AI analyses, ensuring that users can focus first on the most significant issues.

SysAssist balances thorough evaluation with practical recommendations through several mechanisms:

1. Providing concrete suggestions for problem resolution rather than just identifying issues
2. Focusing on issues that have the greatest impact on system quality
3. In the future we would like to extend the capabilities to summarize the findings into overall recommendations to give a more general advice to the users

The system's interface is designed to prevent overwhelming users with excessive findings while still ensuring thorough analysis. When analysing requirements, for instance, the tool not only identifies issues but also suggests improved requirement text, making it easy for users to implement corrections.

5. Case Studies and Validation

5.1 Applications in Real Projects

Our AI-assisted tool has been successfully employed in several significant projects:

1. **EPS-Aeolus Requirements Review**: The tool supported preparation for the EPS-Aeolus requirements review by analysing requirements against INCOSE criteria, allowing the team to address quality issues before formal review.
2. **CO2M Critical Design Review**: As a preparation the CO2M Critical Design Review, the system analysed design specifications against requirements, helping identify areas requiring additional attention and closing gaps
3. **Operational Planning Tool (OPS) Requirements**: The tool reviewed requirements for a weekly operational planning tool enabling scheduling for different missions, identifying issues according to established criteria.

These real-world applications demonstrate the tool's ability to provide value across different phases of the systems engineering lifecycle and in various types of projects.

5.2 Experiences and lessons learned

Our implementation journey involved significant experimentation with various models. Initially, we utilized smaller models and attempted to compensate for their limitations through extensive prompt engineering, which worked partially. However, we ultimately concluded that a robust and capable base model is essential for achieving reliable results. We are currently using the following Qwen2.5-32b-instruct model.

Copyright 2025 by EUMETSAT. Published by the Canadian Space Agency (CSA) on behalf of SpaceOps, with permission and released to the CSA to publish in all forms.

Prompt templates proved valuable for standardizing interactions and ensuring consistent outputs across different analyses. The evolution of these templates has been a continuous process informed by practical application. We developed specialized prompt structures for different use cases, such as a comprehensive document analysis prompt for Technical Notes from a Requirements Engineering perspective.

We established a target of at least 90% correctness in AI assessments, recognizing that lower accuracy would jeopardize user trust and adoption. There are many prompt engineering techniques and approaches, for example Chain-of-thought[5]. Next to these are also prompt templates that give a certain structure to the prompt itself, such as telling the AI to act as a certain persona (i.e. "You are a world-class Systems Engineer with over 20 years of experience"), what style to use, which format and other aspects of the prompt. We have experimented with a lot of them. Our experience is that these prompt engineering techniques help to improve and standardize the results, but the main factor remains the overall quality of the LLM that is used. However, we have the following process to improve the prompt for our specific use cases (example for Use Case 1):

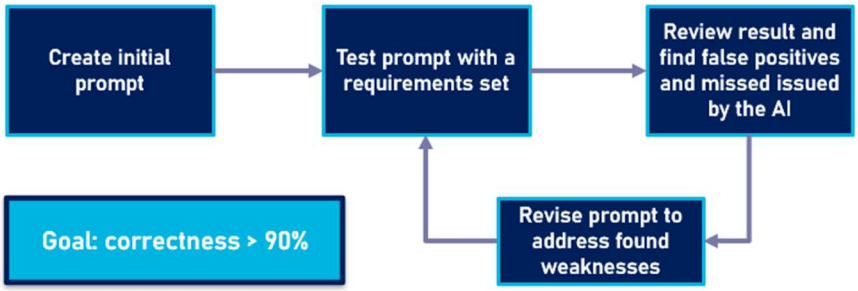


Fig. 7. Prompt Engineering process

To achieve this, we implemented:

- Iterative prompt engineering, extending prompts to address specific issues when false findings were identified
- Utilization of tools like Claude Prompt Improver [4] to refine prompt effectiveness
- Regular validation of findings against expert reviews

The maturity of our use cases varies, with *Use Case1: Requirements Review* being the most developed through multiple iterations and prompt refinements. Our experience shows that with proper tuning, the system can achieve the target reliability levels for most common systems engineering tasks.

Regarding saving of effort, we have come with the following estimation based on feedback received so far:

Use Case	Traditional Method (hrs)	With SysAssist (hrs)	Time Savings (%)
Requirements review (100 requirements)	8.33	5	40
Design adequacy assessment (100 requirements)	10.83	n/a	n/a
Allocation and Flow-Down Analysis (100 requirements)	8.33	6.66	20

Use Case 1 – Requirements review: From experience that checking a requirement against the criteria presented, an effort of 5 minutes per requirement is needed. With SysAssist, the reviewers can simply check if they agree with the assessment made by the AI, which in our experience takes at most 3 minutes per requirement.

Copyright 2025 by EUMETSAT. Published by the Canadian Space Agency (CSA) on behalf of SpaceOps, with permission and released to the CSA to publish in all forms.

Use Case 2 - Design adequacy assessment: The reviewer would normally read the requirement (half a minute) and then read the section in the design document (assumption on average 5 minutes). Finally, the reviewer must think about if the design description really addresses the requirement or not (1 minute). With AI, the reviewer must cross-check the findings of the AI results, which takes less time, since only the findings must be cross checked. If we assume findings on 20% of the requirements, only 20% of effort is needed, but it is not possible to generalize this to an overall estimation. Additionally, the biggest gain in this use case is in the improvement of quality of the design, because not all reviewers do a design adequacy assessment on this level of granularity due to the amount of time it would take. Often, the reviewers do spot-check or focus on their area of expertise only.

Use Case 3 - Allocation and Flow-Down Analysis: Whether all children requirements are fulfilling the parent requirement requires the engineer to read the parent and all child requirements and then check the following questions: 1) can all child requirements be derived or justified by the parent? 2) does the sum of the child requirement cover all aspects of the parent requirement? To do this assessment we would normally need 3 minutes to read the requirements and then to analyse these questions (another 2 minutes). With the help of AI, we can save time on the analysis part (1 minute), since we need to read the requirements anyway to make sure that there are no false findings by the AI.

We received very positive feedback on providing the analysis results to the system engineers giving them actionable insights on how the requirements and the design might be improved or how implicit assumptions were highlighted by the AI to be made more explicit. The phenomenon is often, that the engineers are so involved in the systems requirements and design that sometimes basic information is forgotten to be specified since this is obvious for them.

During exchange within our organization there is a lot of support for the initiative while there is caution who should be provided access to the tools. There is consensus with our approach to provide the tool to requirement authors rather than reviewers to prevent proliferation of duplicate findings. The continued importance of human reviews is recognized for critical systems and should therefore not be compromised.

One key lesson learned was the importance of model quality. While prompt engineering can improve results from smaller models, there is a clear correlation between model capabilities and output quality. As more advanced models have become available, the system's effectiveness has improved significantly.

We have also identified the need for an internal benchmarking process to evaluate new models and prompts/agent, that should ideally be automated based on the human feedback it receives.

6. Challenges and Future Directions

6.1 Current Limitations

Some use cases present inherent challenges, even for human experts. Many findings involve interpretation and engineering judgment rather than clear-cut correctness, making Systems Engineering not always an exact science.

Our initial use of smaller models, which were limited in their capabilities, affected output quality. We've observed that model quality directly correlates with output reliability and usefulness, driving our move toward more advanced models over time. Another limitation we face is the amount of time that is needed by colleagues to evaluate the AI results of our use cases as this is not their main occupation. We will try to incorporate easy feedback mechanisms for the users to mitigate this.

Another challenge is the need for infrastructure that can host bigger LLMs at EUMETSAT internally, since a lot of the data needs to stay internal. We have enough resources for Development and testing but running these use cases as an operational service requires more resources that must be organized and procured first. There are ongoing projects in the organization to address these issues and provide infrastructure to host AI solutions on-premises.

6.2 Additional Future Use Cases

We've identified several promising future use cases that would help the system engineers in our organization:

Test Case Generation from Requirements:

Copyright 2025 by EUMETSAT. Published by the Canadian Space Agency (CSA) on behalf of SpaceOps, with permission and released to the CSA to publish in all forms.

- Based on existing test cases for similar requirements.
- Utilizing RAG (Retrieval Augmented Generation) approaches.
- Following a workflow that includes extracting fitting requirements with related test cases, generating new test cases, and review/improvement cycles.

Advanced Requirements Quality Analysis:

- Enhanced by RAG to incorporate organizational knowledge.
- Providing more context-aware evaluations taking the other requirements into consideration as general knowledge. This could also include the detection of duplicates and inconsistencies.

Deriving Child Requirements from Parent Requirements (Allocation and flow-down):

- Automated support for requirements decomposition from a parent requirement.
- Maintaining consistency with parent requirements.

Document Chat and Analysis:

- Supporting interactive queries about technical documents.
- Extracting specific information from large document sets.

Generic System and Ground Segment requirements:

- Identify differences and commonalities of different System and Ground Segment requirements to tailor them for each individual mission.
- Derive a generic set of System and Ground Segment requirements that can be re-used for each mission.

All these potential use cases require effort to implement and need to be prioritized based on the organizational needs. Therefore, we are always trying to build SysAssist with an architecture that enables reuse to be more efficient to implement these future use cases.

6.3 Agent-Based Approaches and When to Use Them

We're actively exploring when agentic approaches work better than single prompts. Our preliminary findings suggest that tasks requiring creativity and multi-step problem-solving benefit most from agentic implementations. As complexity increases, the value of having multiple specialized agents collaborating on a solution becomes more apparent. Another challenge is choosing the right framework for agents as there are many agentic frameworks and approaches available, for example as described here [6].

Future agent-based systems will include specialized agents for different tools that are in use at EUMETSAT and repositories, including:

- Document Management agent (retrieve related documents) – to retrieve documents from the Document Management system
- Requirements Management agent – to retrieve requirements and traceability information for requirements
- System Modelling agent – for retrieving data from SysML models
- General Systems Engineering assistant agents – to give general guidance and best practices on Systems Engineering

These agents will require an orchestration layer to coordinate their activities and provide a unified interface to users. We are currently implementing first prototypes with agents and are gathering first experiences. Therefore, at this stage it is too early to describe lessons learned.

6.4 Roadmap for Future Development

Our development roadmap focuses on advancing existing use cases with Retrieval-Augmented Generation (RAG) and agentic workflows. We recognize that providing the right context and a clear task definition to the LLM are critical to produce optimal results. Enhancing context management will be a primary focus of our ongoing development.

The roadmap includes:

Copyright 2025 by EUMETSAT. Published by the Canadian Space Agency (CSA) on behalf of SpaceOps, with permission and released to the CSA to publish in all forms.

- Refining existing use cases with improved prompt engineering taking user feedback into consideration directly.
- Adding RAG capabilities to incorporate organizational knowledge. This means that having basic knowledge about documents at EUMETSAT or systems are included automatically.
- Developing specialized agents and tools for different interacting with systems engineering and document management tools.
- Establishing proper infrastructure for production deployment. This is both an organizational and technical task to facilitate the usage of the AI tools like SysAssist in a productive environment.

7. Conclusion

SysAssist represents an important contribution to the practice of Systems Engineering at EUMETSAT through several key innovations:

1. Development of a user-friendly AI tool that makes advanced AI capabilities accessible to systems engineers without requiring specialized AI knowledge
2. Successful application in real projects like EPS-Aeolus and CO2M, demonstrating practical value
3. Time savings ranging from 10-30% in verification and validation tasks, based on our requirements engineering team's experience using the tool across multiple review cycles.
4. An innovative approach of providing AI tools to requirement authors rather than reviewers, maximizing quality improvements early in the development process
5. A flexible architecture that accommodates both on-premises processing for sensitive data and cloud-based processing for enhanced performance

The impact of these contributions extends beyond mere automation, enabling a fundamental shift in how systems engineers work by allowing them to focus on creative and analytical tasks while AI handles routine verification activities. This shift promises not only efficiency gains but also improvements in overall system quality through more comprehensive verification.

As AI technologies continue to mature, we anticipate increasingly sophisticated support for Systems Engineering processes, ultimately enabling engineers to focus on high-value creative and analytical tasks while routine verification activities are increasingly automated.

The future of Systems Engineering will likely involve a hybrid approach where AI and human engineers work collaboratively, each leveraging their unique strengths. SysAssist represents an early step in this direction, demonstrating the potential of AI to enhance rather than replace human expertise and judgement in the complex domain of Systems Engineering. At organizational level, we have multiple AI related activities in the System Engineering area and we will integrate and standardize the solutions that are being implemented.

Acknowledgements

I would like to thank EUMETSAT and my immediate management to support these AI activities and I would also like to thank the colleagues from the Generative AI working group to collaborate and give feedback to this work.

References

List of references

[1] GovTech Singapore. (2023). Prompt Engineering Playbook (Beta v3). Data Science & Artificial Intelligence Division, Government Technology Agency of Singapore. Retrieved from https://www.developer.tech.gov.sg/products/collections/data-science-and-artificial-intelligence/playbooks/prompt-engineering-playbook-beta-v3.pdf

[2] INCOSE Guide for Writing Requirements. INCOSE-TP-2010-006-04| VERS/REV:4, 2023

[3] Anthropic. Improve your prompts in the developer console. November 14 2024, from https://www.anthropic.com/news/prompt-improver, (accessed 06.03.25).

Copyright 2025 by EUMETSAT. Published by the Canadian Space Agency (CSA) on behalf of SpaceOps, with permission and released to the CSA to publish in all forms.

[4] ECSS-M-ST-10C Rev.1 (2009). "Space project management: Project planning and implementation." European Cooperation for Space Standardization.
[5] Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., & Zhou, D. (2022). Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. arXiv preprint arXiv:2201.11903
[6] Anthropic. (2024). Building effective agents. Retrieved from https://www.anthropic.com/research/building-effective-agents (accessed 12.03.25)
[7] Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., & Chen, W. (2021). LoRA: Low-Rank Adaptation of Large Language Models. arXiv preprint arXiv:2106.09685.
[8] Ollama. (2024). Ollama: Get up and running with Llama 3.3, DeepSeek-R1, Phi-4, Gemma 3, and other large language models. Retrieved from https://github.com/ollama/ollama/blob/main/docs/api.md (accessed 14.03.25)
[9] text-generation-webui (2024) - A Gradio web UI for Large Language Models – Retrieved from https://github.com/oobabooga/text-generation-webui (accessed 14.03.25)