Proc. of the European Safety and Reliability Conference (ESREL 2026)
Edited by José C. Matos, Paulo B. Lourenço, Michael Beer, Daniel Oliveira,
Edoardo Patelli, Hélder S. Sousa, Rui A. Silva and Monica Santamaria
 Published by Research Publishing, Singapore.
doi: 10.3850/ESREL2026061419_esrel26-p26509-cd

AI Agent for Assessing the Quality of FMEA Reports

Jean Meunier-Pion

Laboratoire de Génie Industriel, CentraleSupélec, Université Paris-Saclay, France.
E-mail: jean.meunier-pion@centralesupelec.fr

Zhiguo Zeng

Laboratoire de Génie Industriel, CentraleSupélec, Université Paris-Saclay, France.
E-mail: zhiguo.zeng@centralesupelec.fr

Anne Barros

Laboratoire de Génie Industriel, CentraleSupélec, Université Paris-Saclay, France.
E-mail: anne.barros@centralesupelec.fr

Anticipating and managing risks in the early development of industrial systems is essential to reduce costs and hazards through design improvements. Widely accepted methods such as Failure Modes and Effects Analysis (FMEA) were designed precisely to support risk prevention during system design. However, conducting an FMEA is costly, and because FMEAs are qualitative analyses that rely on natural language, using Large Language Models (LLMs) to automate parts of this work is increasingly attractive. Yet, ensuring the quality of such AI-generated analyses remains challenging. Currently, FMEAs are mainly crafted and assessed by human experts, while too few automated methods exist to evaluate their quality. Meanwhile, the "LLM-as-a-judge" paradigm has emerged, where LLM agents evaluate task outputs. In this work, we: (1) propose a method to automatically generate FMEA reports, (2) introduce a LLM-as-a-judge framework to automatically produce review reports that assess FMEA quality, and (3) compare AI-generated and human-generated FMEAs using both our LLM-as-a-judge framework and human evaluations.

Keywords: Failure Modes and Effects Analysis (FMEA), Large Language Model (LLM), LLM-as-a-judge, Evaluation, FMEA generation

1. Introduction

1.1. *Failure Modes and Effects Analysis*

Failure Modes and Effects Analysis (FMEA) is a widely used method in quality and reliability engineering for conducting risk assessment early in system design (Wu et al. (2021)). It aims to identify early on potential failure modes, analyze their causes and effects, and document supporting information such as existing controls and detection methods, severity/occurrence/detection ratings, the resulting Risk Priority Number (RPN) used to rank risks, and recommended actions to reduce or mitigate the identified risks.

The purpose of conducting an FMEA is to identify potential failure modes as early as possible, enabling design improvements that increase system reliability. FMEA is also an almost-mandatory requirement in multiple industries where safety is a primary concern, such as automotive industry, space industry, defense industry (AIAG and VDA (2021); Duphily (2009); United States Department of Defense (1980)). And in several other industries, such as medical devices, civil aviation, or rail transport, FMEA is frequently used as a deliverable for mandatory risk assessment, although FMEA is not mandatory in itself.

1.2. *Limitations of FMEA*

The current way FMEAs are conducted comes with several limitations. The way FMEA knowledge is written down makes it hard to compare, aggregate, or reuse across products, sites, and teams, because FMEA content is often semi-structured, with ad-hoc field naming, which limits integration and knowledge reuse across tools and lifecycles (Hodkiewicz, Melinda et al. (2021); Wu

et al. (2021); Razouk, Houssam et al. (2023)). Moreover, classic FMEA rarely capture the rationale: who proposed what; based on which evidence; why a particular wording or score was chosen; etc. This makes design rationale and traceability difficult (Hodkiewicz, Melinda et al. (2021); Wu et al. (2021)). Besides, FMEA is time- and resource-intensive because it is meeting-driven. It is traditionally conducted by teams of 6-10 human experts, including a facilitator, engineers, suppliers, end-users, that gather to discuss and identify relevant information to include in the analysis (AIAG and VDA (2021); Tavčar and Duhovnik (2014)). Depending on the system, FMEA can in general take a hundred of staff hours, which represents a non-negligible workforce cost.

1.3. *Related work and research gap*

1.3.1. *Advanced informatics for FMEA*

The use of AI and NLP to normalize language and structure semantics is a promising way to make FMEA entries comparable, searchable, and reusable, while aiming at reducing the time needed to conduct FMEA, as well as enabling more exhaustive and tracaeable analyses (Razouk, Houssam et al. (2023)).

Several approaches have already been proposed to help practitioners conduct FMEA with the assistance of AI or ontologies, but fully automated, end-to-end FMEA remains uncommon in industrial practice and still challenging in research prototypes (Wu et al. (2021)). Ontology-based FMEA aims to formalize domain knowledge so that failure knowledge can be queried, reused, and partially generated automatically. While this improves consistency and supports reuse within a well-bounded domain, it typically requires significant manual knowledge engineering, careful maintenance when products evolve, and it struggles with open-ended, organization-specific wording and tacit expert knowledge that is rarely captured in a reusable form (Hodkiewicz, Melinda et al. (2021)).

More recently, LLM-based FMEA generation has emerged as an attractive alternative because it can produce plausible FMEA entries directly from natural-language documents (El Hassani, Ibtissam et al. (2025); Lynch, Karol aet al. (2024)). Yet, LLM-generated FMEAs remain difficult to operationalize without human oversight due to well-known issues such as confabulation, missing or inconsistent assumptions, unstable terminology, and limited traceability: the model can generate fluent entries without them being relevant to the specific system under study (Collier, Zachary A. et al. (2025); Diemert and Weber (2023)).

1.3.2. *Evaluation of FMEA*

A second barrier to automation concerns evaluation: even when an AI system produces an FMEA table, determining whether it is correct, complete, and useful is not straightforward. Most existing works evaluate FMEA through small case studies, expert judgement, or user studies, often using ad-hoc criteria and limited reporting of datasets, prompts, and rating procedures. As a result, comparisons across methods are difficult and reproducibility is limited. Automated checks (e.g., presence of required columns/sections, consistency of RPN calculations, or formatting constraints) can improve reliability, but they do not capture semantic quality, such as whether a failure mode is specific, whether causes and effects are plausible for the given component, or whether recommended actions meaningfully reduce risk. One of the main challenge of evaluation being that FMEA is mostly a text generation task, is motivates a complementary evaluation approach leveraging an LLM-as-a-judge agent, designed to produce structured review reports that combine compliance checks with content-focused critique (Gu, Jiawei et al. (2025)).

1.4. *Proposed approach and contributions*

This paper addresses this gap by proposing (i) a framework for FMEA generation that can be fully AI-driven or conducted through human/AI interaction, and (ii) an AI agent framework that automatically reviews FMEA reports to support evaluation of such analyses.

The proposed contributions in this paper are the following:

(1) A framework for automatically generating

FMEA using an LLM.
(2) An LLM-based evaluation agent to assess FMEA compliance.
(3) Detailed observations on FMEA content quality and recommended solutions.

In Sect. 2, we introduce FMEA and define the scope of our study. Then, in Sect. 3, we describe our proposed FMEA generation approach using LLM. Sect. 4 presents our approach for evaluating FMEA. In Sect. 5, we show and discuss results obtained on a case study conducted on an Armpi FPV robot arm. We conclude the paper with Sect. 6 with future perspectives for LLM generation and evaluation.

2. Problem statement

2.1. *Addressed tasks*

An FMEA comes traditionally in the form of a table summarizing the potential failure modes of a system, their causes, effects, and other information. This table comes in general as part of a broader report, describing the process and the conclusions related to this table.

In this work, we mainly address two tasks: **(1) FMEA generation**, and **(2) FMEA evaluation**. The second task can be subdivided into two subtasks: (2.1) FMEA compliance evaluation, and (2.2) FMEA content evaluation. Task (1) deals with automatically generating an FMEA, given some input data about a given system. Task (2) deals with evaluating the quality of an FMEA report. Subtask (2.1) consists of verifying that a given report is compliant with the FMEA standard it is said to follow, while subtask (2.2) evaluates the intrinsic quality of the FMEA content.

2.2. *Scope of the study*

Here, we restrain ourselves to the military FMEA standard MIL-STD-1629A, and apply it to an Armpi FPV robot arm.

Armpi FPV robot arm – This system comprises six servo motors to enable the movement of a robotic arm. To narrow down the study, we focus on one of the servo motors of this system, the LX-225 servo motor, which serves at the base of the robot arm. We chose this system because we had already some knowledge and experience with it, while it also represented an example of a small complex system, making it convenient and manageable for a toy FMEA case study.

MIL-STD-1629A – We consider MIL-STD-1629A for the FMEA standard in this work because it provides a clear, structured and actionable list of elements that should appear in an FMEA report, making it convenient to measure the compliance of reports.

3. FMEA generation

Our end goal is to generate high-quality FMEAs (semi-)automatically. To this end, we first propose, in this section, an FMEA generation framework that automatically produces FMEA reports.

3.1. *Overall generation process*

We define an FMEA generation agent that follows 12 steps to produce an FMEA report (Figure 1). Starting from the system description, the agent identifies components and their functions, then derives failure modes, causes, and effects. It then generates detection methods and assigns severity, occurrence, and detection ratings, from which the RPN is computed. Finally, it assembles the generated entries into an FMEA table and formats them into a report compliant with the target FMEA standard.

3.2. *User input data: system description*

The user should provide the AI agent, as primary input data, a system description report and the name of the system. Additional data can also be ingested such as images, additional documents, and user notes. However, for conciseness, we did not use such data sources for this work and keep them as prototypes for future research.

The system description report currently contains description of the operating conditions of the system, its scope, functional requirements, and comes with a list of hardware components and software modules.

3.3. *Additional context: web data*

In addition, we allow the generation agent to access the Internet through a search engine API to retrieve search results and scrape web pages relevant to each task, in order to enrich the input context. More specifically, for each generation block (*e.g.*, the failure modes block), we allow the agent to query up to five search results (a limit set as a trade-off to obtain decent results from the search API) using a query template, the system name, and keywords related to the block (*e.g.*, "failure modes"). Search results can have variable quality, especially when they point to webpages returning 4xx/5xx errors (*e.g.*, error 404) or websites protected by CAPTCHAs. To validate the quality of collected webpages, we use a two-step process: first, a heuristic based on the minimum length of extracted webpage text; second, a verification agent prompted to filter out webpages that do not contain sufficiently insightful content.

Ideally, in the future, the generation agent should have access to at least the three following databases: (1) the Internet, (2) the FMEA-generation-service provider database, and (3) the end-user local database.

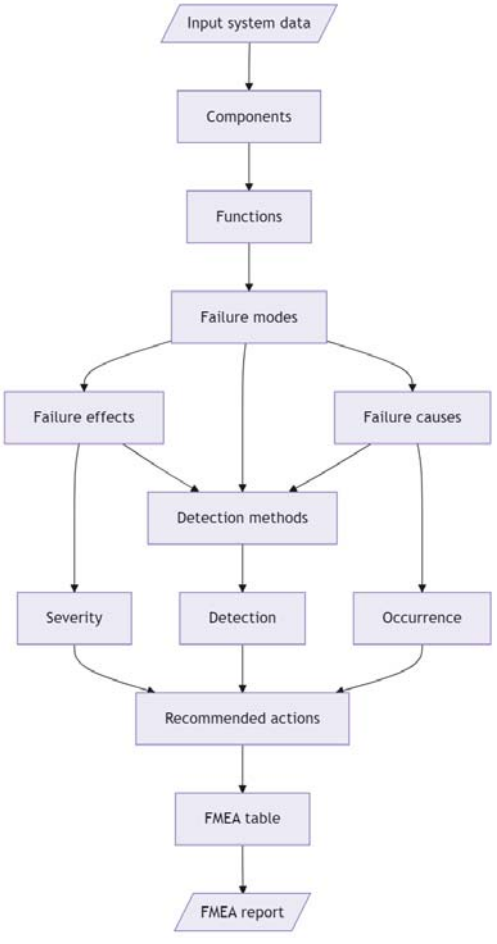

Fig. 1. Flowchart of the overall generation process.

3.4. *Block-level generation process*

At block level, all blocks follow the same generation process (Figure 2). From the system name (used for web retrieval), system description, and optional additional inputs, we build a block-specific prompt. The prompt is sent to an LLM, and the response is parsed into a JSON file containing the generated items. For all LLM tasks in this work, we use GPT-4o-mini through the OpenAI API as a trade-off between performance and cost. The quality of generated failure modes primarily depends on prompt design, including block-specific rules and few-shot examples that clarify expectations for FMEA entries. We also apply a taxonomy to map raw LLM outputs to standardized final outputs. In addition, we provide a chat-based AI assistant to edit intermediate outputs through predefined actions ("add element", "remove element", "add description", "edit element description", "edit element name", "edit references"); users can also perform these edits manually.

3.5. *Report edition*

All blocks contribute to crafting the FMEA table, except the last one which consists of piecing together all data to write the final FMEA report. For this step, several additional pieces of information must be provided by the user. Those include mainly assumptions for the FMEA, such as how does the user define failure or severity levels, etc.

4. FMEA evaluation

A key challenge in evaluating FMEAs is that comparing an FMEA against a "gold FMEA" is

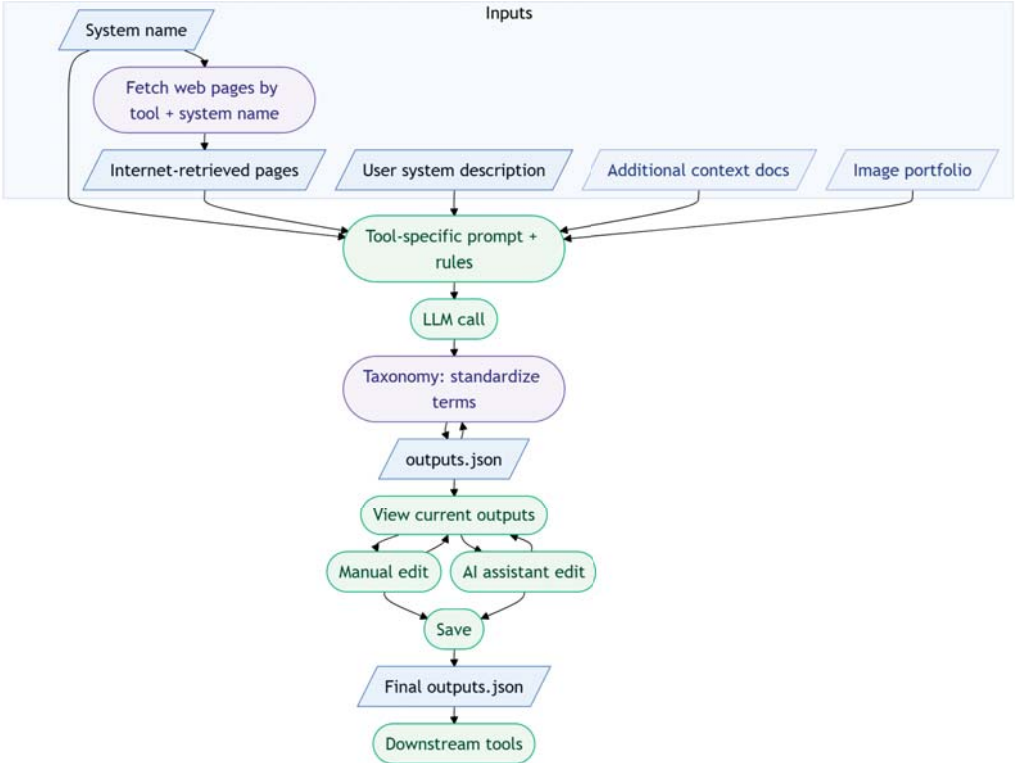

Fig. 2. Flowchart of the block-level generation process.

neither viable nor scalable, for at least two reasons: (1) producing ”gold FMEA” data is costly, and there are too few high-quality public ”gold FMEAs”; and (2) there is no single perfect FMEA for a given system, a limitation shared with text-generation evaluation in general.

4.1. *Compliance evaluation*

An FMEA is a risk-assessment deliverable governed by specific standards. Beyond semantic relevance, an FMEA should follow the structure prescribed by a standard. Therefore, we first assess FMEA quality through compliance with the selected FMEA standard. For LLM-based frameworks, where generated outputs can appear convincing while remaining too vague or insufficiently specific to be actionable, compliance evaluation helps ensure structure correctness, thus preventing fuzzy or missing assumptions from undermining high-quality FMEA generation.

The MIL-STD-1629A standard is a military FMEA standard which provides a set of elements that an FMEA report should contain. We implement compliance evaluation as a set of 34 automated checks ensuring that the elements required by MIL-STD-1629A are covered and organized as stated by the standard. The 34 checks are grouped into six main sections:

(1) **Report structure** (presence and relative order of required FMEA report sections),
(2) **System description** (e.g., presence of system diagrams),
(3) **Standard and method** (reference to the standard and risk calculation method used),
(4) **Scope, ground rules and assumptions** (scope of the analysis),
(5) **Essential elements of the FMEA** (presence of FMEA table and essential columns),
(6) **Summary** (major problems, recommenced actions, and omitted items).

First, to verify the report structure, we use a regular expression to collect Markdown headings and their associated body text into a list of sections. Then, for each expected section in the FMEA standard, we identify the most similar section in the evaluated report and match them. As title phrasing can vary across reports, we use an LLM instead of regular expressions for matching sections.

Then, we run the other compliance checks. (See an example compliance check prompt in Figure 3.) Each check consists of an instruction, the criterion to check, some context extracted from the report, and the expected output format. A check score in [0, 1] can then be automatically parsed from the prescribed structured format.

We aggregate the individual check scores to obtain a global compliance score. We define compliance score to be the average of all passed checks (see Eq. (1)).

$$\text{Compliance}(r, c) = \sum_{i=1}^{\#c} \text{check}(r, c_i) \qquad (1)$$

Where r is an FMEA **r**eport and c a finite set of checks to be applied to r. $\text{check}(r, c_i)$ returns 1 if the LLM outputs a score s greater than a threshold of 0.6 (arbitrary choice), and 0 if it is less than the threshold.

We compare two approaches for checking compliance: (1) multi-prompt approach, which consists of running one prompt per check, and (2) single-prompt approach, which consists of running one large prompt for all checks at once.

4.2. *Content evaluation*

Beyond FMEA compliance, evaluating FMEA content is crucial for assessing how relevant the generated risk analysis is. Furthermore, an automated method for evaluating FMEA content would enable rapid iterations to further improve FMEA-generation models.

In this work, most effort have been put into the generation and compliance evaluation parts. For a matter of conciseness, we decide to restrain content evaluation to qualitative human observations in Sect. 5, discussing future perspectives as well as empirical improvements implicitly made in this work.

5. Results and discussion

5.1. *Case study*

We evaluated four different FMEAs: (1) one written by a human annotator, (2) a naive GPT-4.1 baseline, (3) one fully-AI-generated using our framework, and (4) one AI/human-generated using our framework.

The GPT-4.1 baseline represents an attempt at generating an FMEA using ChatGPT naively, while the human-written FMEA serves as a model of how to follow the MIL-STD-1629A standard.

5.2. *Compliance evaluation results*

For compliance evaluation, we provide the obtained results in Table 1.

Table 1. Compliance results

FMEA	Compliance Multi-prompt	Compliance Single-prompt
Human	**80.6** $\pm$ **5.8** %	**96.7** $\pm$ **1.0** %
Full AI	78.2 $\pm$ 1.4 %	84.3 $\pm$ 6.3 %
AI & Human	79.4 $\pm$ 0.0 %	78.3 $\pm$ 4.1 %
GPT 4.1 raw	48.1 $\pm$ 1.4 %	41.7 $\pm$ 2.9 %

Since LLMs can introduce randomness with output tokens sampling, we provide compliance score results across 15 runs for each FMEA. We report "mean $\pm$ standard deviation".

The human baseline obtained the higher compliance scores. A likely explanation is that it contains compliance-oriented content the checker rewards, such as assumptions, severity classes, and occurrence levels, whereas those checks often fail for the AI-generated FMEAs, meaning that such specifications could be better defined in AI-generated reports.

The GPT-4.1 baseline exhibited the lower compliance scores by a significant margin. In fact, such FMEA reports generated from an AI assistant like ChatGPT, without careful guidance, miss essential methodological elements such as clear definitions and scope for the conducted analysis. This leads to many compliance checks failing.

Even though the "Full AI" and "AI & Human" reports share the same structure and only differ

Prompt

```
Instruction
You are evaluating the 'System Description' section of an FMEA report.

Criterion
Does it describe expected performance at system/subsystem/component levels?

Text (input section)
The HiWonder Armpi_FPV is a 6-degree-of-freedom robotic arm designed primarily for
educational purposes, enabling educators and students to explore key robotics
concepts including manipulation and computer vision. Controlled by a Raspberry Pi
4B running ROS 1 Melodic, the system supports functionalities such as object
recognition, manipulation, and remote operation via PC or smartphone interfaces.

Key subsystems of the Armpi_FPV include the robotic arm assembly, integrated camera
 system for visual feedback, and the communication module that connects to external
 devices. It interacts with users through Wi-Fi and internet protocols, operating
as a ROS node that sends and receives commands from a ROS master.

Operating in indoor environments, the robotic arm is suitable for demonstrations
and hands-on learning. It is important to note that the system is not engineered
for industrial applications due to its low torque capabilities, which limit its
operational capacity. Assumptions include a stable power supply and a conducive
environment for computer vision tasks, such as adequate lighting and minimal
obstacles.

Output format constraint
Return ONLY JSON: {"score": <float in [0,1]>, "justification": "<brief rationale
(1-2 sentences)>"}
```

Model output

```
{
  "score": 0.9,
  "justification": "The text provides a clear description of the expected
  performance of the system and its subsystems, detailing functionalities and
  limitations, which aligns well with the criterion."
}
```

Fig. 3. Example of compliance check prompt.

in terms of FMEA table contents and the analysis of the failure modes, their compliance scores exhibit differences, particularly for the single-prompt strategy. A reason could be that the FMEA content itself influences the compliance scores, although they should rely primarily on the report structure and the presence of methodological elements. Besides, the observed variability of compliance scores, especially for the single-prompt strategy, suggests the compliance evaluator should be tightened to ensure robust assessment of FMEA quality.

Although it is still challenging to draw definite conclusions, we note that our current generation framework enables fully- and semi-automated FMEAs to comply almost on-par with human-written FMEAs, while a naive FMEA generation using ChatGPT does not successfully comply and may lack the structure and key assumptions of a well-done FMEA.

5.3. *Human observations on content*

We report multiple key human observations on the FMEAs contents. Some of those observations have already been leveraged to improve the FMEA generation process, while the others will serve as key knowledge to iterate on future versions of the FMEA generation and evaluation frameworks.

First, using LLMs naively and without clearly specifying what we expect from an FMEA cannot lead to a satisfactory output. We indeed ob-

served several instances of erratic behavior, including: outputs being too vague and not system-specific or component-specific; outputs being too component-specific and not relevant to the system as a whole (*e.g.*, failure modes concerning servo motor electronic circuit or gears, instead of failure modes of the servo motor as a component of the system); (near-)duplicate outputs; generation of many entries thus overflowing the FMEA.

This makes the case for stressing: the need for specific outputs; the targeted indenture levels at each FMEA step; the use of a taxonomy to standardize text and handle duplicates; the need for conciseness. Moreover, LLMs tend to mix failure modes, causes, and effects, and specific prompt engineering needs to be done to ensure that failure modes do not include causality phrases.

We also noted differences in the way the FMEA contents are considered between the human-generated FMEA and the ones from the framework. Indeed, the human one relied more on physical phenomena to describe failure modes (*e.g.*, "overheating", "motor demagnetization"), while our framework is more prone to generating failure modes driven by functional requirements (*e.g.*, "Failure to stop joint movement on command").

An advantage of LLM-generated FMEA over traditional approaches is that each generated entry can come with citations from reference documents that can then be documented in specific appendices, allowing for easier traceability and auditability. However, extracting relevant citations from reference documents is still challenging.

6. Conclusion

This work is part of an ongoing research on automating FMEA generation using LLMs. We proposed a first FMEA generation framework, along with a compliance evaluator, and provided key findings on LLM caveats for generating FMEAs.

Next steps for this research include (1) enhancing explainability of the generation process, by extracting relevant citations from references and providing LLM explicit reasoning to support the generated FMEAs, and (2) designing, implementing, and validating a robust and systematic FMEA content evaluator.

Acknowledgement

The authors gratefully acknowledge the financial support of the chair on Risk and Resilience of Complex Systems (RRCS) for this research.

References

AIAG and VDA (Eds.) (2021). *FMEA-Handbuch* (1. Ausgabe, korrigierter Nachdruck ed.). Berlin: VDA.

Collier, Zachary A. et al. (2025, April). How good are large language models at product risk assessment? *Risk Analysis 45*, 1–20.

Diemert, S. and J. H. Weber (2023, July). Can Large Language Models Assist in Hazard Analysis? arXiv.

Duphily, R.J. and Command, A. (2009). Space vehicle failure modes, effects, and criticality analysis (FMECA) guide. Technical report, El Segundo, CA, USA.

El Hassani, Ibtissam et al. (2025). AI-driven FMEA: integration of large language models for faster and more accurate risk analysis. *Proc. Des. Soc.*.

Gu, Jiawei et al. (2025, November). A Survey on LLM-as-a-Judge.

Hodkiewicz, Melinda et al. (2021, September). An ontology for reasoning over engineering textual data stored in FMEA spreadsheet tables. *Computers in Industry 131*, 103496.

Lynch, Karol aet al. (2024). FMEA Builder: Expert Guided Text Generation for Equipment Maintenance.

Razouk, Houssam et al. (2023, December). Improving FMEA Comprehensibility via Common-Sense Knowledge Graph Completion Techniques. *Expert Systems with Applications 25*, 121367.

Tavčar, J. and J. Duhovnik (2014). Tools and Methods Stimulate Virtual Team Co-operation at Concurrent Engineering. pp. 457–466. IOS Press.

United States Department of Defense (1980). MIL-STD-1629A.

Wu, S., Z. Li, J. Liu, and X. Xu (2021). Literature review and prospect of the development and application of FMEA in manufacturing industry. *Int J Adv Manuf Technol 112*, 1957–1970.