Forschungsbericht 2026-26

Development and Evaluation of a GenAI-based Methodology for Aircraft Systems Engineering

Viola Voth

Deutsches Zentrum für Luft- und Raumfahrt
Institut für Flugsystemtechnik
Braunschweig

DLR Deutsches Zentrum für Luft- und Raumfahrt
German Aerospace Center

Forschungsbericht 2026-26

Development and Evaluation of a GenAI-based Methodology for Aircraft Systems Engineering

Viola Voth

Deutsches Zentrum für Luft- und Raumfahrt
Institut für Flugsystemtechnik
Braunschweig

298 Seiten
53 Bilder
83 Tabellen
231 Literaturstellen

DLR **Deutsches Zentrum für Luft- und Raumfahrt**



Herausgeber:

Deutsches Zentrum
für Luft- und Raumfahrt e.V.
Wissenschaftliche Information
Linder Höhe
D-51147 Köln

ISSN 1434-8454
ISRN DLR-FB-2026-26
Erscheinungsjahr 2026
DOI: 10.57676/ks0g-bn28

Erklärung des Herausgebers
Dieses Werk – ausgenommen anderweitig gekennzeichneter Teile – ist lizensiert unter den Bedingungen der Creative Commons Lizenz vom Typ Namensnennung 4.0 International (CC BY 4.0), abrufbar über https://creativecommons.org/licenses/by/4.0/legalcode.

Lizenz

Creative Commons Attribution 4.0 International

Generative Künstliche Intelligenz (GKI), Große Sprachmodelle (LLM), Retrieval-Augmented Generation (RAG), Systems Engineering, Flugzeugsystementwicklung, Wissensextraktion

Viola VOTH
DLR, Institut für Flugsystemtechnik, Braunschweig

Entwicklung und Evaluierung einer auf generativer KI basierenden Methodik für den Entwurf von Flugzeugsystemen
Technische Universität Braunschweig

Die Entwicklung neuer Flugzeugmodelle und -varianten ist mit hohen Kosten und langen Entwicklungszeiten verbunden. Als wesentlicher Faktor hierfür ist die Vielzahl vernetzter Systeme zu nennen. Deren Entwicklung folgt etablierten Systems-Engineering-Methodiken. In der industriellen Praxis erfolgt zunächst eine Prüfung, welche Elemente wiederverwendet werden können und welche angepasst oder neu entwickelt werden müssen. Dieser Prozess ist hochgradig iterativ und ressourcenintensiv. Die vorliegende Arbeit untersucht die Frage, inwiefern generative KI dazu beitragen kann, diese frühe Entwicklungsphase zu beschleunigen.

Die im Rahmen dieser Arbeit entwickelte SAGE-Methodik gliedert sich in vier Phasen: Datenaufbereitung, Wissensextraktion, Wissenssynthese und LLM-gestützter Entwurf bei neuen Anforderungen. Anhand zweier Fallstudien, dem Environmental Control System und dem Thermal Management System, wurden implizite Informationen aus öffentlich zugänglichen Quellen systematisch extrahiert und aufbereitet. Dabei zeigte sich, dass aus fragmentierten Quellen vollständige und konsistente Systemrepräsentationen erzeugt werden können. Die generierten Architekturen sind konzeptionell, ingenieurtechnisches Fachwissen bleibt unverzichtbar. Damit positioniert sich der entwickelte Ansatz als Katalysator initialer Entwicklungsentscheidungen mit Übertragbarkeit auf weitere technische Domänen.

Generative Artificial Intelligence (GenAI), Large Language Models (LLM), Retrieval-Augmented Generation (RAG), Systems Engineering, Aircraft Systems Design, Knowledge Extraction
(published in english)

Viola VOTH
German Aerospace Center (DLR), Institute of Flight Systems, Braunschweig

Development and Evaluation of a GenAI-based Methodology for Aircraft Systems Engineering
Technical University of Braunschweig

Developing new aircraft models and variants is costly and time-consuming. A key factor in this regard is the multitude of interconnected systems. Their development follows established systems engineering methodologies. In industrial practice, this involves first assessing which established elements can be reused, adapted, or must be newly developed. This process is highly iterative and resource-intensive. This dissertation investigates how generative AI can accelerate this early development phase.

The developed methodology SAGE comprises four phases: data preparation, knowledge extraction, knowledge synthesis, and LLM-supported design for new requirements. Based on two case studies, the Environmental Control System and the Thermal Management System, implicit information was systematically extracted and processed from publicly accessible sources. The results showed that complete and consistent system representations can be generated from fragmented sources. The generated architectures remain conceptual, and engineering expertise remains indispensable. This positions the developed approach as an accelerator for early-stage development decisions, with transferability to other technical domains.

TU Braunschweig - Niedersächsisches
Forschungszentrum für Luftfahrt

Berichte aus der Luft- und Raumfahrttechnik

Forschungsbericht 2026-18

Development and Evaluation of a GenAI-based Methodology for Aircraft Systems Engineering

Viola Voth

Deutsches Zentrum für Luft- und Raumfahrt
Institut für Flugsystemtechnik
Braunschweig

Diese Veröffentlichung wird gleichzeitig in der Berichtsreihe „NFL - Forschungsberichte“ geführt.

Diese Arbeit erscheint gleichzeitig als von der Fakultät für Maschinenbau der Technischen Universität Carolo-Wilhelmina zu Braunschweig zur Erlangung des akademischen Grades einer Doktor-Ingenieurin genehmigte Dissertation.

Development and Evaluation of a GenAI-based Methodology for Aircraft Systems Engineering

Von der Fakultät für Maschinenbau
der Technischen Universität Braunschweig
zur Erlangung der Würde

einer Doktor-Ingenieurin (Dr.-Ing.)

genehmigte Dissertation

von:	Viola Voth
geboren in:	Hamburg

eingereicht am:	27.11.2025
mündliche Prüfung am:	17.06.2026

Vorsitz:	Prof. Dr.-Ing. Jens Friedrichs
Gutachter:	Prof. Dr.-Ing. Stefan Levedag
Gutachter:	Prof. Dr.-Ing. Christoph Herrmann
Gutachter:	Prof. Ph.D. Jon Wade

2026

Acknowledgments

This dissertation would not have been possible without the support, encouragement, and contributions of many people, to whom I owe my deepest gratitude.

First and foremost, I wish to express my sincere thanks to Prof. Dr.-Ing. Stefan Levedag for his competent and dedicated supervision of my doctoral studies. The valuable technical discussions and constructive suggestions for possible topics significantly contributed to the successful completion of this work. I am also very grateful to Prof. Dr.-Ing. Christoph Herrmann for kindly taking on the role of examiner and for his genuine interest in my research. Likewise, I am thankful to Prof. Dr.-Ing. Jens Friedrichs for serving as chair of the examination committee.

My sincere thanks go to Prof. Ph.D. Jon Wade (University of California, San Diego) for his inspiring mentorship and unwavering belief in my work. You taught me that it's sometimes necessary to test the limits of a system to understand its capabilities. This lesson has profoundly shaped both my research and my thinking. Your encouragement made my time at UCSD an exceptionally enriching experience, and this dissertation would not have taken shape in its present form without your influence. Thank you for your enthusiasm and for the many discussions that broadened my perspective far beyond engineering.

I would like to thank Dr.-Ing. Andreas Bierig and Dr.-Ing. Oliver Bertram for their ongoing support, valuable discussions, and constructive feedback throughout my time at the German Aerospace Center (DLR). Thanks to you, I had the opportunity to spend five months conducting research in the United States. An experience that continues to shape my professional and personal development.

My gratitude also goes to my colleagues at the DLR for their support, countless discussions, and productive teamwork. In particular, I would like to mention Sascha Lübbe, Michael Schäfer, Florian Jäger, and Alexander Bismark for the many professional and personal conversations and for sharing their passion for aircraft engineering. Thank you!

I am also thankful to Dr.-Ing. Robert Doering and Morris Jaensch from Airbus for the great collaboration within the SKAIB project. Even though our project results are not directly part of this dissertation, working with you was both inspiring and motivating.

Special thanks go to the students who contributed to this dissertation, as well as to the UCSD Master's class of 2025 in Architecture-Based Enterprise Systems Engineering, for their warm welcome and for making my stay in San Diego such an inspiring time.

I am especially indebted to those who took the time to proofread and comment on parts of this dissertation: Prof. Ph.D. Jon Wade, Sascha Lübbe, Dr.-Ing. Oliver Bertram, Sören Voth, and Hans-Bodo von Ditfurth. Your feedback, precision, and attention to detail have significantly enhanced the quality of this work.

Finally, I would like to express my sincere gratitude to my family and friends for their unwavering support and encouragement. I am especially grateful to my brother Sören for sharing his profound insights on AI and for always being there with advice and support whenever I needed it. My heartfelt thanks go to my partner, Hans-Bodo, for his unconditional support and patience. Thank you for always believing in me, even when we were thousands of kilometers apart. You encouraged me to keep going when things became difficult, and your presence has been my biggest support throughout this journey. Above all, I am very grateful to my parents, who nurtured my curiosity, fostered my love for science, and supported me every step of the way. Without your support and encouragement, I would not be where I am today.

This work is dedicated to all of you.

Braunschweig, November 2025
Viola Voth

Abstract

Developing new aircraft models and variants is costly and time-consuming. In addition to organizational factors, the multitude of interconnected systems (e.g., Environmental Control System or Anti-Ice System) is particularly responsible for this. Their development follows established systems engineering methodologies, in which architectures are specified step by step from requirements through functions and logical structures to physical components. Methodologically, this process could be carried out entirely from scratch. In industrial practice, however, engineers first assess which established elements can be reused, adapted, or must be newly developed. This so-called Reuse/Adapt/New analysis is highly iterative, repetitive, and resource-intensive.

This dissertation examines how Generative Artificial Intelligence (GenAI) can accelerate this process. The methodological focus is on integrating Large Language Models (LLMs) with Retrieval-Augmented Generation (RAG), which combines LLMs' semantic analysis capabilities with structured access to domain-specific knowledge. Building on this, the SAGE (*System Architecture through Generative Engineering*) methodology is developed. It consists of four phases: (1) structured data preparation, (2) recursive knowledge extraction, (3) knowledge synthesis for reconstructing existing systems using the RFLP logic (Requirements, Functions, Logical, Physical), and (4) LLM-supported development of new system designs based on new boundary conditions. The last phase encompasses the Reuse/Adapt/New analysis, the complete RFLP design, and the creation of consistent system architectures that serve as starting points for downstream optimization.

The methodology's feasibility is verified through two case studies: the Environmental Control System (ECS) and the Thermal Management System (TMS) for fuel cell integration. Based solely on publicly accessible scientific publications and patents, information was systematically extracted, structured, and prepared for architecture development. The results showed that complete and consistent system representations can be generated from fragmented sources, providing a solid foundation for downstream development decisions.

The results show both potential and limitations. SAGE can efficiently index large document sets, reconstruct complex system relationships, and structure them coherently along RFLP logic. This is much faster than manual methods. Limitations appear in mathematical modeling, detailed optimization, and final architectural development. The generated architectures are consistent yet conceptual. Engineers' expertise remains indispensable. The potential lies in combining automated knowledge processing with expert evaluation.

Overall, this dissertation demonstrates the transformative potential of the developed methodology for aircraft engineering while simultaneously opening perspectives for transferability to other technical domains. SAGE positions itself as a powerful accelerator in the early development phase by laying the foundation for faster, more informed engineering decisions.

Zusammenfassung

Die Entwicklung neuer Flugzeugmodelle bzw. -varianten ist mit hohen Kosten und langen Entwicklungszeiten verbunden. Neben organisatorischen Faktoren sind hierfür insbesondere die Vielzahl vernetzter Systeme (z.B. Environmental Control System, Anti-Ice-System, etc.) verantwortlich. Deren Entwicklung folgt etablierten Systems-Engineering Methodiken, bei denen Architekturen schrittweise von Anforderungen über Funktionen und logischen Strukturen bis hin zu physischen Komponenten spezifiziert werden. Methodisch könnte dieser Prozess zwar vollständig von *scratch* erfolgen, in der industriellen Praxis wird jedoch zu Beginn geprüft, welche etablierten Elemente wiederverwendet werden können und welche angepasst oder neu entwickelt werden müssen. Diese sogenannte Reuse/Adapt/New Analyse ist hochgradig iterativ, repetitiv und ressourcenintensiv.

Diese Dissertation untersucht, wie Generative Künstliche Intelligenz (GenAI) zur Beschleunigung dieses Prozesses beitragen kann. Im Zentrum steht die Kombination von Large Language Models (LLMs) mit Retrieval-Augmented Generation (RAG). Diese verbindet die semantischen Analysefähigkeiten großer Sprachmodelle mit einem strukturierten Zugang zu domänenspezifischem Wissen. Darauf aufbauend wird die SAGE-Methodik ("*System Architecture through Generative Engineering*") entwickelt. Diese besteht aus den vier Phasen: (1) strukturierte Datenaufbereitung, (2) rekursive Wissensextraktion, (3) Wissenssynthese zur Rekonstruktion bestehender Systeme entlang der RFLP-Logik (Requirements, Functions, Logical, Physical) sowie (4) der LLM-gestützten Entwicklung neuer Systemdesigns auf Basis neuer Randbedingungen. Die letzte Phase umfasst die Reuse/Adapt/New-Analyse, die vollständige RFLP-Entwicklung und die Generierung konsistenter Systemarchitekturen als Ausgangspunkt für nachgelagerte Optimierungen.

Die Machbarkeit der Methodik wird anhand zweier Fallstudien überprüft: dem Environmental Control System (ECS) und dem Thermal Management System (TMS) für die Integration von Brennstoffzellen. Auf Basis ausschließlich öffentlich zugänglicher wissenschaftlicher Publikationen und Patente wurden implizite Informationen systematisch extrahiert, strukturiert und für die Architekturentwicklung aufbereitet. Dabei zeigte sich, dass aus fragmentierten Quellen vollständige und konsistente Systemrepräsentationen erzeugt werden können, die eine fundierte Grundlage für nachgelagerte Entwicklungsentscheidungen bilden.

Die Ergebnisse zeigen sowohl Potenziale als auch Grenzen der Methodik auf. Die Methodik kann umfangreiche Dokumentensammlungen systematisch erschließen, komplexe Systemzusammenhänge rekonstruieren und diese entlang der RFLP-Logik konsistent strukturieren. Für diese Aufgaben sind manuelle Verfahren erheblich zeitaufwändiger. Grenzen zeigen sich hingegen bei der detaillierten mathematischen Modellierung, der quantitativen Optimierung und der Ausarbeitung finaler Systemarchitekturen. Die von SAGE erzeugten Architekturen stellen konsistente, aber konzeptionelle Ausgangspunkte dar. Hier bleibt

die Expertise der Ingenieure weiterhin unverzichtbar. Das größte Potenzial liegt in der Arbeitsteilung zwischen automatisierter Wissenserschließung und ingenieurwissenschaftlicher Bewertung.

Insgesamt zeigt diese Dissertation das transformative Potenzial der entwickelten Methodik für die Luft- und Raumfahrttechnik und eröffnet zugleich Perspektiven für eine Übertragbarkeit auf andere technische Domänen. SAGE positioniert sich als leistungsstarker Katalysator in der frühen Entwicklungsphase, indem es die Grundlage für schnellere und fundiertere Ingenieursentscheidungen schafft.

List of Publications

Source	Article	Relevant chapters
[1]	Viola Voth, Sascha M. Lübbe, Oliver Bertram: "Estimating Aircraft Power Requirements: A Study of Electrical Power Demand Across Various Aircraft Models and Flight Phases". In: *Aerospace* 11(12) (2024). DOI: 10.3390/aerospace11120958	1, 6, A.14, A.15
[2]	Viola Voth, Oliver Bertram: "Aircraft System Design: A Model-Based and Collaborative Approach". In: *International Council of the Aeronautical Spaces*. 2024	2
[3]	Viola Voth, Sascha Lübbe, Michael Schäfer, Axel Berres, Oliver Bertram: "Functional Approach to a Fuel Cell Thermal Management System in Safety-Critical Applications". In: *AIAA AVIATION 2023 Forum*. 2023. DOI: 10.2514/6.2023-3879	6, A.2
[4]	Sascha Lübbe, Viola Voth, Michael Schäfer et al.: "Interconnections in Model-Based Safety Analysis and Systems Design on the Example of a Fuel Cell Thermal Management System for Commercial Aircraft". In: *AIAA AVIATION 2023 Forum*. 2023. DOI: 10.2514/6.2023-4196	/
[5]	Markus Schorr, Viola Voth, Christoph Gentner: "Effects on the design of aeronautical fuel cell systems by inclusion of reliability requirements". In: *CEAS Aeronautical Journal*, 15(4) (2024), pp.807-820. DOI: 10.1007/s13272-024-00743-9	A.14
[6]	Markus Schorr, Viola Voth, Christoph Genter: "Auswirkungen von Redundanzanforderungen auf den Entwurf von Brennstoffzellen-Systemen in Verkehrsflugzeugen". In: *Deutscher Luft- und Raumfahrtkongress*. 2022	A.14
[7]	Viola Voth, Jon Wade, Oliver Bertram, Andreas Bierig: "Evaluation of a Generative AI Methodology for Technical System Development: A Case Study of an Aircraft Environmental Control System". In: *AIAA SciTech 2026 Forum*. 2026.	2.4, 3, 5

Usage of Digital Tools

The following AI tools were used to assist in the preparation of this dissertation:

Translation tools:

- DeepL for translating texts from German to English[1]
- Anthropic Claude 3.7 Sonnet[2] for isolated 1:1 translations of specific text passages (with explicit prompts for literal translation)

Text improvement:

- Grammarly[3] and Anthropic Claude 3.7 Sonnet[2] for checking and improving grammar, spelling, and style
- DeepL Write[4] for stylistic improvement and optimization of text passages

These tools were used exclusively as aids for linguistic improvements and translations and served as initial support. All texts translated or improved by AI tools were then critically reviewed and revised by the author. All technical content, analyses, interpretations, and scientific findings originate from the author. The use of these tools has no influence on the originality of the scientific work and the research results presented therein.

[1] https://www.deepl.com/de/translator
[2] https://claude.ai/new
[3] https://app.grammarly.com/
[4] https://www.deepl.com/de/write

Table of Contents

List of Figures xv

List of Tables xix

Nomenclature xxi

1. Introduction 1
- 1.1. Problem Statement . . . 3
- 1.2. Research Assumptions and Questions . . . 6
- 1.3. Thesis Structure and Reading Guide . . . 8
- 1.4. Summary . . . 10

2. State of the Art 11
- 2.1. Document-Based Systems Engineering . . . 11
  - 2.1.1. The V-Model . . . 12
  - 2.1.2. Challenges of Document-based Systems Engineering . . . 14
- 2.2. Model-Based Systems Engineering . . . 15
  - 2.2.1. Theoretical Foundation and Core Principles . . . 15
  - 2.2.2. Advantages of MBSE . . . 17
  - 2.2.3. Implementation Barriers . . . 18
- 2.3. Development of Civil Aircraft and their Systems . . . 19
  - 2.3.1. Aircraft Development Procedure . . . 19
  - 2.3.2. Aircraft System Development Procedure . . . 21
  - 2.3.3. Aircraft Development as a Knowledge-Intensive Domain . . . 21
- 2.4. Artificial Intelligence . . . 23
  - 2.4.1. AI Technologies . . . 23
  - 2.4.2. Large Language Models . . . 25
  - 2.4.3. Retrieval-Augmented Generation . . . 33
  - 2.4.4. Literature Analysis: RAG and LLMs in Aircraft Engineering . . . 43
- 2.5. Summary . . . 45

3. Development of the SAGE Methodology 47
- 3.1. Phase 1: Data Preprocessing . . . 48
  - 3.1.1. Data Collecting . . . 48
  - 3.1.2. Data Structure of Scientific Publications and Patents . . . 51
  - 3.1.3. Data Parsing & Cleaning of Textual Document Elements . . . 54
  - 3.1.4. Data Chunking, Indexing, and Vectorization . . . 60
- 3.2. Phase 2: Extended RAG with Recursive Knowledge Extraction . . . 62
  - 3.2.1. Categorized Initial Questions . . . 63
  - 3.2.2. Retrieval Process . . . 65
  - 3.2.3. Answer Generation . . . 66
  - 3.2.4. Recursive Knowledge Exploration via Follow-up Questions . . . 66
  - 3.2.5. Knowledge Integration and Storage . . . 68
- 3.3. Phase 3: LLM Knowledge Synthesis . . . 68
  - 3.3.1. Knowledge Synthesis . . . 69
  - 3.3.2. Evaluation of the Knowledge Synthesis Report . . . 70

3.4. Phase 4: LLM Architecture Generation . . . 71
3.4.1. Architecture Generation Process . . . 71
3.4.2. Evaluation of the LLM-developed Architecture . . . 72
3.5. Discussion and Summary . . . 74

4. Empirical Analysis of the Proposed Methodology 75
4.1. System Parameters: Classification and Influence Analysis . . . 75
4.1.1. Data Preprocessing Parameters . . . 75
4.1.2. Parameters of the Extended RAG-System . . . 78
4.1.3. Parameters of Knowledge Synthesis and Architecture Generation . 80
4.2. Systematic Parameter Studies . . . 82
4.2.1. Study A: Vector Store Size and Number of Answered Queries . . . 82
4.2.2. Study B: Analysis of the k-Value . . . 85
4.2.3. Study C: Comparison of Different LLM Models for Answer Generation 87
4.2.4. Study D: Influence of Prompt Length on Report Quality . . . 91
4.2.5. Study E: Evaluating LLM Models for Knowledge Synthesis . . . 95
4.3. Limitations and Future Optimization Potential . . . 99
4.4. Summary . . . 99

5. Case Study 1: Environmental Control System 101
5.1. Use Case: Environmental Control System . . . 101
5.1.1. Conventional Bleed-Air ECS . . . 101
5.1.2. Bleed-free ECS (eECS) . . . 103
5.2. Phase 1: Data Preprocessing . . . 104
5.2.1. Scope of Processed Data . . . 104
5.3. Phase 2: Extended RAG . . . 105
5.3.1. High-Level Questions for Knowledge Synthesis . . . 106
5.3.2. Statistical Analysis of Question-Answer Generation . . . 108
5.4. Phase 3: Knowledge Synthesis Through LLM Integration . . . 110
5.4.1. Prompt Engineering for Knowledge Synthesis . . . 111
5.4.2. LLM Configuration and Settings . . . 112
5.4.3. Knowledge Synthesis Report Generation . . . 112
5.4.4. Evaluation of Knowledge Synthesis Report . . . 114
5.5. Phase 4: Architecture Generation Through LLM Reasoning . . . 118
5.5.1. Prompt Engineering for Architecture Generation . . . 118
5.5.2. LLM Configuration and Settings . . . 120
5.5.3. Architecture Generation Report . . . 120
5.5.4. Evaluation of the Architecture Generation Capability . . . 125
5.6. Summary . . . 128

6. Case Study 2: Thermal Management System 129
6.1. Use-Case: Thermal Management System . . . 129
6.1.1. System Architecture and Boundary Conditions . . . 130
6.1.2. Assumed Aircraft Architecture . . . 130
6.1.3. Assumed Fuel Cell System Architecture . . . 131
6.2. Phase 1: Data Preprocessing . . . 133
6.2.1. Scope of Processed Data . . . 133
6.3. Phase 2: Extended RAG . . . 134
6.3.1. High-Level Questions for Literature Exploration . . . 134
6.3.2. Statistical Evaluation of Question-Answer Generation . . . 135
6.4. Phase 3: LLM Knowledge Synthesis . . . 136
6.4.1. Prompt Engineering for Knowledge Synthesis . . . 138
6.4.2. LLM Configuration and Settings . . . 139
6.4.3. Result of the Knowledge Synthesis . . . 139
6.4.4. Qualitative Evaluation of Knowledge Synthesis . . . 142

6.5. Phase 4: LLM Architecture Generation . . . 143
6.5.1. Prompt Engineering for Architecture Generation . . . 143
6.5.2. LLM Configuration and Settings . . . 144
6.5.3. Architecture Generation . . . 145
6.5.4. Qualitative Evaluation of the Generated Architecture . . . 148
6.6. Summary . . . 148

7. Discussion **151**
7.1. Iteration Time and Workload . . . 151
7.2. Classification of SAGE within Existing Development Methodologies . . . 152
7.3. Advantages and Limitations of the SAGE Methodology . . . 153
7.4. Industry Impact . . . 154

8. Conclusion and Outlook **155**

Bibliography **159**

A. Appendix **177**
A.1. Definitions of Technical Terminology . . . 177
A.2. ARP Guidelines for Aircraft System Development . . . 182
A.3. Historical Development of AI . . . 184
A.4. Extraction and Analysis of Visual Document Elements . . . 186
A.5. Prompt for the LLM-based Knowledge Extraction . . . 191
A.6. Prompts for the Evaluation . . . 192
A.7. Modular Prompt for the System Context Generation . . . 195
A.8. High-Level Questions for the Development of the ECS . . . 196
A.9. Prompt Example of the ECS Case Study . . . 198
A.10. Representative Excerpts from ECS Knowledge Synthesis Report . . . 204
A.11. Evaluation of the ECS Knowledge Synthesis Report . . . 216
A.12. Representative Excerpts from ECS Architecture Generation Report . . . 222
A.13. Evaluation of the Architecture Generation Variants . . . 228
A.14. Fuel Cell Systems in Aviation . . . 237
A.15. Calculation of the Boundary Conditions for the Assumed FuCS Architecture 248
A.16. High-Level Questions for the TMS Development . . . 258
A.17. Representative Excerpts from TMS Knowledge Synthesis Report . . . 260
A.18. Representative Excerpts from TMS Architecture Generation Report . . . 265

List of Figures

1.1. Aircraft technology evolution from the mechanical to the big data era, illustrating the rise in measured parameters and passenger numbers, accompanied by a reduction in cockpit crew size (1930-2020). Simplified representation based on publicly available data about typical aircraft. . 2
1.2. Principle of a classical RAG system: user queries are enriched with retrieved context from structured knowledge bases before being processed by the LLM, adapted from [43]. 5
1.3. Structure of the dissertation 9

2.1. RFLP framework in the V-model, adapted from [44, p.158] 13
2.2. Visionary system model vs. SysML system model, adapted from [44, p.55] 15
2.3. Overview of existing diagram types in SysML v1, adapted from [58, p.13] 16
2.4. Airbus' V-model for aircraft development, adapted from [80, p.27], [85, p.7] 20
2.5. Representation of the extended V-model for multi-level development [2] 22
2.6. Hierarchical categorization of GenAI and LLMs within the AI spectrum, adapted from [90] 23
2.7. Development timeline of selected LLMs from three major providers: Anthropic, OpenAI, and Google, from 2018 to 2025 (own representation based on the data from Thompson [97]) 25
2.8. Schematic overview of the functionality of an LLM 26

2.9. Architectural overview of a typical RAG system, adapted from [133] . . 34
2.10. Illustration of a semantic vector space in a RAG system. Each point represents a text chunk, and spatial proximity reflects semantic similarity. 35
2.11. Visualization of vector-based similarity search with different distance metrics 36
2.12. Overview of the multi-hop QA concept 40
2.13. Overview of the iterative concept 41
2.14. Overview of the concept of structured approaches 42

3.1. Overview of the different phases in the SAGE methodology 47
3.2. Schematic structure of the data preprocessing pipeline, extended in this work with an explicit *Data Structure Analysis* step to reflect the characteristics of scientific publications and patents. 49
3.3. Schematic activity diagram of keyword-based data acquisition 50
3.4. Schematic overview of the structural characteristics of a scientific article (left) [173] and a patent document (right) [174] 52
3.5. Flow diagram of a PDF preprocessing pipeline that analyzes different document types and extracts, processes, and combines content into a common text data format using specific parsers 54

3.6. Data preprocessing workflow for scientific publications using the first page of a fuel cell cooling system research paper by Lee and Kim [173]. Left: Original PDF file with layout-based formatting. Middle: Parsed text in Markdown format with preserved structure. Right: Cleaned text version after the cleaning process with optimized readability and removed formatting metadata. For better readability, the image has been rotated by 90°. 59
3.7. The conventional retrieval process 63
3.8. Retrieval process flowchart: A comprehensive visualization of the two-path retrieval system, showing both the hierarchical question exploration path (left) and the direct answer generation path (right). 64
3.9. Retrieval workflow implemented in this dissertation 65
3.10. Growth of question count across recursion levels with an assumption of 2 follow-up questions per guiding question and 30 guiding questions 67
3.11. Knowledge synthesis process 69
3.12. Process of the system architecture generation process 71

4.1. Configurable parameters of the methodology across the four phases . . 76
4.2. Percentage of answered questions by category (left) and hierarchy level (right) for different vector store configurations. Average answer coverage: 72% (20 TMS Papers), 85% (All TMS Papers), 93% (Complete Literature). Categories: SC = System Context, RQ = Requirements, FU = Functions, LE = Logical Elements, PC = Physical Components, AP = Architecture Patterns. 84
4.3. Percentage of answered questions by category (left) and hierarchy level (right) for different k-values. Average answer coverage: 73.7% at $k = 5$, 86.5% at $k = 10$, 93.2% at $k = 20$, 95.9% at $k = 40$. 86
4.4. Excerpt of synthesized HTML report generated by GPT-4.1-mini . . . 97
4.5. Final system implementation showing the selected parameter values for the four main operational phases 100

5.1. Schematic structure of the ECS system in the A320 102
5.2. Schematic structure of the eECS system in the B787, adapted from [192, p.1374] 103
5.3. Answered ECS questions by category (left) and by hierarchy level (right). Blue bars represent the total number of questions analyzed, while red bars denote the subset that could be answered with the information available in the database (yes) 108
5.4. Knowledge synthesis process 111
5.5. Architecture generation process of the eECS 118
5.6. System architecture of variant 1 (left), variant 2 (right), and the reference system of the B787 (bottom) 124

6.1. Assumed power supply aircraft architecture with fuel cell power supply 131
6.2. Assumed FuCS system architecture, adapted from [206] 132
6.3. Answered TMS questions by category (left) and hierarchy level (right). Blue bars show all analyzed questions, while red bars denote the subset that could be answered with the information available in the database (Answerable). 135
6.4. Developed TMS architecture (reconstructed by the author from LLM-generated tabular specifications in Table 6.7). 146

AF.1. Dependency between the system development and the safety assessment, adapted from [28, p.32] 182
AF.2. Flowchart of the LLM-supported document analysis for extracting and processing images and associated text. 187
AF.3. Comparison between schematic ECS architecture (left) and physical ram air installation (right) used for automated image interpretation experiments. Pictures from [193] 188
AF.4. Source utilization analysis showing LLM selection from available QA-pairs across knowledge categories and hierarchical levels, demonstrating strategic information extraction for comprehensive synthesis 217
AF.5. Schematic structure of a PEM fuel cell, adapted from [200, p.29], [203, p.119] 239
AF.6. Schematic progression of a voltage-current density characteristic curve, adapted from [203, p.19] 240
AF.7. Schematic progression of a i-A curve with the various loss mechanisms occurring during operation, adapted from [203, p.205] 241
AF.8. Schematic system architecture of liquid cooling (left) and evaporative cooling (right), adapted from [199, p.15] 247
AF.9. Polarization curve of the fuel cell stack “P Stack” [231], adjusted. . . . 250

List of Tables

2.1. Benchmark results comparing GPT-o1 and GPT-4.1 with data from [114] 29

3.1. Evaluation criteria for parser comparison 55
3.2. Comparison of different parsers 57

4.1. Classification of system parameters and their methodological influence 81
4.2. Performance metrics of various LLMs in the RAG system 89
4.3. Comparison of requirements coverage by category 93
4.4. Excerpt of the HTML document that was generated with the monolithic prompt, showing the external interfaces of the ECS 93
4.5. Excerpt of the HTML document that was generated with the modular prompt, showing the external interfaces of the ECS 93
4.6. Comparison of analyzed LLM models regarding their context window. Data from [186]. 95
4.7. Excerpt of synthesized HTML table generated by GPT-4.1 96
4.8. Excerpt of synthesized HTML table generated by Gemini-2.0-Flash . . 97

5.1. Overview of answered questions by category 108
5.2. Overview of answered questions by hierarchy level 109
5.3. ECS external system interfaces (example of knowledge synthesis report) 112
5.4. Key ECS functional requirements (example of knowledge synthesis report) 113
5.5. ECS function hierarchy (example from knowledge synthesis report) . . 113
5.6. Overall traceability assessment results 117
5.7. Excerpt of the component reusability matrix from the report 121
5.8. Excerpt of the external system boundary definition from the report . . 122
5.9. Architecture 1: single-stage HPC component analysis (representative sample) 123
5.10. Architecture 2: two-stage HPC component analysis (representative sample) 123

6.1. Overview of answered questions by category 135
6.2. Overview of answered questions by hierarchy level 136
6.3. TMS synthesis report - functional requirements hierarchy analysis . . . 140
6.4. Excerpt of the TMS function inventory - primary functions 141
6.5. Excerpt of the TMS logical elements registry 141
6.6. Excerpt of logical elements mapped to physical TMS components . . . 145
6.7. TMS generation report - final validated system design specification . . 147

AT.1. High-Level Questions of the ECS Case Study 196
AT.2. External systems and interfaces (verbatim from synthesis report) . . . 204
AT.3. Interfaces: Parameters, protocols, and formats (verbatim from report) . 205
AT.4. System environment and operating conditions (verbatim from report) . 205
AT.5. Stakeholder requirements and expectations (verbatim from report) . . . 205

AT.6. ECS operating modes (verbatim from report) . . . 206
AT.7. Impact of flight phases on ECS (verbatim from report) . . . 206
AT.8. Complete listing of external interfaces (verbatim from report) . . . 207
AT.9. Quantitative parameters for each interface (verbatim from report) . . . 207
AT.10. ECS synthesis report - excerpt of the performance requirements list . . 208
AT.11. ECS synthesis report - excerpt of the functional requirements list . . . 209
AT.12. Continuation of the functional requirements list . . . 210
AT.13. ECS synthesis report - function hierarchy and classification . . . 210
AT.14. Main ECS Functions (excerpt, simplified) . . . 211
AT.15. ECS synthesis report - logical component hierarchy and classification . 212
AT.16. ECS synthesis report - physical component hierarchy and classification 213
AT.17. ECS synthesis report - physical component hierarchy and classification (Part 2) . . . 214
AT.18. Excerpt of ECS Functional Decomposition (simplified) . . . 215
AT.19. Excerpt of ECS Physical Architecture View (simplified) . . . 215
AT.20. Source fidelity distribution of synthesized statements . . . 216
AT.21. Overall traceability assessment results . . . 220
AT.22. ECS generation report - component reusability matrix . . . 223
AT.23. ECS generation report - external system boundary definition . . . 224
AT.24. ECS generation report - system environment and constraints . . . 225
AT.25. ECS generation report - stakeholder needs analysis . . . 225
AT.26. ECS generation report - key functional requirements . . . 226
AT.27. ECS generation report - function identification and hierarchy (excerpt) 227
AT.28. ECS generation report - logical element identification (excerpt) . . . 227
AT.29. Quantitative assessment of prompt execution quality dimensions . . . 229
AT.30. Quantitative assessment of mathematical accuracy and feasibility . . . 231
AT.31. Quantitative assessment of technical understanding dimensions . . . 234
AT.32. Quantitative assessment according to the elegant design framework . . 235
AT.33. Overview of fuel cell types and their characteristics [203, p.8], [205, p.7] 238
AT.34. Comparison of selected hydrogen storage forms [228, p.18], [205, p.7] . 243
AT.35. Environmental Parameters for Various Flight Conditions . . . 249
AT.36. Fuel Cell Stack Parameters for Various Flight Phases . . . 250
AT.37. Heat Flows [kW] for Various Flight Phases . . . 255
AT.38. TMS Interface Parameters . . . 256
AT.39. Assumed mission data . . . 257
AT.40. TMS synthesis report - functional requirements . . . 260
AT.41. TMS synthesis report - comprehensive function registry . . . 260
AT.42. TMS synthesis report - function hierarchy and dependencies analysis . 262
AT.43. TMS synthesis report - cross-domain function applicability matrix . . . 262
AT.44. TMS synthesis report - logical elements registry . . . 263
AT.45. TMS synthesis report - traceability matrix - requirements to elements . 264
AT.46. TMS synthesis report - comprehensive component registry (excerpt) . . 264
AT.47. TMS generation report - functional requirements extraction . . . 265
AT.48. TMS generation report - performance requirements for aircraft case . . 266
AT.49. TMS generation report - function registry for aircraft application . . . 266
AT.50. TMS generation report - function hierarchy for aircraft application . . 267
AT.51. TMS generation report - logical elements registry for aircraft application 267
AT.52. TMS generation report - logical elements to physical components mapping 269
AT.53. TMS generation report - thermodynamic analysis (hybrid configuration) 270
AT.54. TMS generation report - pressure drop validation analysis . . . 271
AT.55. TMS generation report - final validated system design specification . . 272

Nomenclature

Abbreviations

Symbol	Description
A/C	Aircraft
AEA	All Electric Aircraft
AI	Artificial Intelligence
AIME	American Invitational Mathematics Examination
AP	Architecture Patterns
API	Application Programming Interface
APU	Auxiliary Power Unit
ARP	Aerospace Recommended Practice
ATA	Air Transport Association
BERT	Bidirectional Encoder Representations from Transformers
CAST	China Academy of Space Technology
CFD	Computational Fluid Dynamics
CONOPS	Concept of Operations
CoT	Chain of Thought
CS	Certification Specifications
DC	Direct Current
DBSE	Document-Based Systems Engineering
DDMS	Digital Design Manufacturing & Service
DOI	Digital Object Identifier
EASA	European Aviation Safety Agency
ECS	Environmental Control System
eECS	Electric Environmental Control System
ESA	European Space Agency
FAA	Federal Aviation Administration
FCOM	Flight Crew Operating Manual
FHA	Functional Hazard Analysis
FU	Function
FuC	Fuel Cell
FuCS	Fuel Cell System

Symbol	Description
FTA	Fault Tree Analysis
GAN	Generative Adversarial Network
GenAI	Generative Artificial Intelligence
GPOA	Graduate-level Physics Questions Assessment
GPT	Generative Pre-trained Transformer
HPC	High-Pressure Compressor
HTML	Hypertext Markup Language
HX	Heat Exchanger
IMA	Integrated Modular Avionic
INCOSE	International Council on Systems Engineering
IP	Intellectual Property
ISO	International Organization for Standardization
JSON	JavaScript Object Notation
LE	Logical Element
LLM	Large Language Model
LTL	Linear Temporal Logic
MBSE	Model-Based Systems Engineering
MEA	More Electric Aircraft
ML	Machine Learning
MMLU	Measuring Massive Multitask Language Understanding
MMMU	Massive Multitask Multimodal Understanding
NASA	National Aeronautics and Space Administration
NIST	Computer Security Resource Center
NLP	Natural Language Processing
OCR	Optical Character Recognition
PASA	Preliminary Aircraft Safety Assessment
PC	Physical Component
PDF	Portable Document Format
PEMFC	Proton Exchange Membrane Fuel Cell
PSSA	Preliminary System Safety Analysis
PTS	Purchaser Technical Specifications
QA	Question-Answer
RAG	Retrieval-Augmented Generation
RE	Requirements Engineering
RBD	Reliability Block Diagram
RFLP	Requirements-Functions-Logical-Physical
RLHF	Reinforcement Learning from Human Feedback
RQ	Requirement

Symbol	Description
SC	System Context
SE	Systems Engineering
SOI	System of Interest
SoS	System of Systems
SRD	System Requirements Document
SSA	System Safety Analysis
SysML	Systems Modeling Language
TLAR	Top-Level Aircraft Requirement
TLSRD	Top-Level System Requirements Document
TMS	Thermal Management System
UML	Unified Modeling Language

Latin Symbols

Symbol	Description	Unit
A_Z	Electrode Area	m^2
C	Context clarity	-
c	Speed of sound	m/s
c_p	Specific heat capacity	kJ/(kg·K)
D	Data	-
d	Number of dataset doublings	-
E	Effectiveness	-
G^0	Molar Free Reaction Enthalpy	kJ/mol
F	Faraday Constant	(A·s)/mol
h	Specific enthalpy	J/kg
H	Altitude	m
I	Electric Current	A
k	Number of retrieved chunks	-
$\dot{m}$	Mass flow	kg/s
Ma	Mach Number	-
n	Number of follow-up questions	-
$\dot{n}_{cell}$	Number of cells	-
$\dot{n}_{El}$	Electron Flow	mol/s
p	Pressure	bar
P_{Stack}	Stack power	W
$\dot{Q}$	Heat	W

Symbol	Description	Unit
Q	Charge Quantity	C
Q_A	Quality of Analysis/Reports	-
RH	Relative humidity	%
S	Scope of the task	-
t	Time	s
T	Temperature	K
U^0	Maximum Theoretical Cell Voltage	V
U	Remaining share of unanswered questions	-
U_0	Initial proportion of unanswered questions for the smallest dataset configuration	-
$\dot{U}_{cell}$	Average cell voltage	V
$\dot{U}_{Stack}$	Stack voltage	V
v	Flight speed	m/s
w	Weighting factor	-
x	Moisture content	g
z	Number of electrons per atom	-

Greek Symbols

Symbol	Description	Unit
η	Efficiency	-
κ	Heat capacity ratio	-
λ	Stoichiometric ratio	-
ξ	Mass fraction	-

1. Introduction

Developing new commercial aircraft requires considerable investment of time and resources. Estimates suggest costs in the tens of billions of US dollars and development times of several years. There are no official cost figures available from the manufacturers, and the available estimates are primarily based on press reports and scientific analyses. Winter [8, p.165] shows that the actual costs of the last three major aircraft programs (A350, A380, and B787) far exceeded the originally planned project volumes. For example, Winter estimates the original development cost for the Airbus 380 at around US$16 billion, while the actual costs are estimated to have risen to over US$22 billion [8, p.165], spanning more than a decade from its initial development in the early 1990s to the first flight in 2005 [9].

On the one hand, the actual project volumes often differ considerably from the original planned values, and on the other hand, development time is frequently exceeded. For instance, it took around nine years to certify the Boeing 787, rather than the originally planned five years [10, p.2]. These deviations demonstrate a close correlation between development time and costs, making shortening development cycles a key objective for the aircraft industry. This underlines the growing importance of methods and tools that enhance development flexibility and enable faster responses to market changes.

At the same time, modern aircraft programs are becoming increasingly complex. An aircraft is a system comprised of a large number of interconnected subsystems. According to ISO 15288 [11, p.11] and INCOSE [12, p.265], a *system* is generally defined as a combination of elements, subsystems, or assemblies whose functionality is greater than the sum of its individual elements. These elements can include "*products (hardware, software, firmware), processes, people, information, techniques, facilities, services, and other support elements*" [12, p.5]. The Airbus 380, for example, has more than 70 (sub-) systems [13, p.1], including the flight control and air conditioning systems.

Within the spectrum of technical systems, it is also important to distinguish between different categories based on their predictability and behavioral characteristics. *Complicated* systems are characterized by thousands of precisely engineered parts with intricate dependencies, whose behavior remains predictable through systematic decomposition. In contrast, *complex* systems develop emergent properties from component interactions, making the prediction of their future behavior inherently difficult even when individual components are well understood[1] [14, pp.3-7].

As illustrated in Figure 1.1, aircraft have evolved over the past century from mechanically simple systems to highly complicated technological systems. The Douglas DC-3, introduced in 1935, could be operated by just two pilots under visual flight rules, carrying up to 32 passengers while monitoring basic parameters such as temperature, altitude, and pressure [15]. As aircraft and their operation became more complicated, the requirements of the flight crew increased accordingly. The Lockheed Constellation, for instance, required between five and eight cockpit crew members, including two pilots, a flight engineer, a

[1] Throughout this dissertation, the distinction between *complicated* and *complex* systems follows the Cynefin framework developed by Snowden and Boone [14, pp.3-7]. However, much of the existing literature uses the term *complex systems* when referring to systems that, according to this definition, are complicated.

navigator, and a radio operator, to manage the increasingly sophisticated systems [16, p.4]. However, improvements in automated systems and digital integration have enabled a return to a two-pilot cockpit despite increased aircraft capacities. Modern aircraft such as the Airbus 380 accommodate over 850 passengers while requiring the same two-person flight crew [9], yet incorporate up to 25,000 sensors throughout their flight systems [17]. A single Boeing 787 flight generates an average of 500 GB of system data, while modern General Electric turbine engines continuously monitor more than 5,000 data points per second [17]. This represents a fundamental shift from manually monitoring basic aircraft parameters to the automated processing of vast data streams encompassing every aspect of aircraft performance and operation.

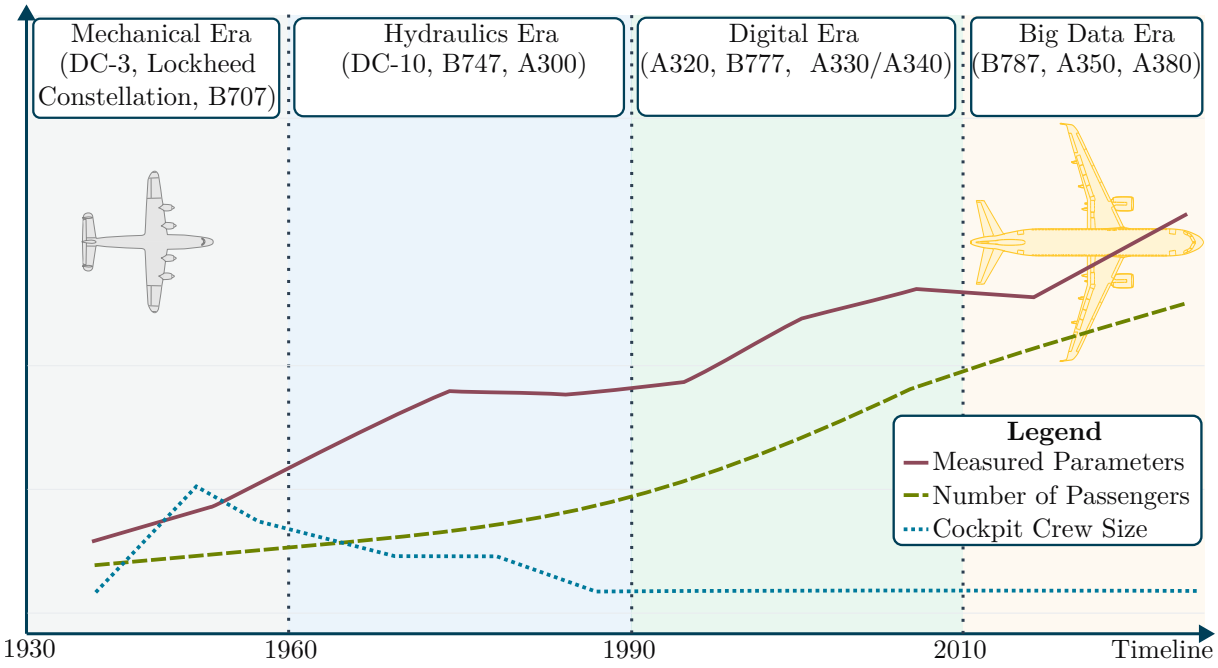

Figure 1.1.: Aircraft technology evolution from the mechanical to the big data era, illustrating the rise in measured parameters and passenger numbers, accompanied by a reduction in cockpit crew size (1930-2020). Simplified representation based on publicly available data about typical aircraft.

While aircraft systems are becoming increasingly complicated from a technical standpoint, completely new developments are becoming less common in the civil aircraft industry. Instead, new programs focus on the systematic reuse and further development of existing systems.

In this dissertation, a genuinely *new development* is defined as the creation of an entirely new technical system that is not based on previous versions and does not adopt concepts from other domains. Such cases are extremely rare in the civil aircraft industry. Even landmark designs such as the Douglas DC-3, which established the concept of the modern commercial aircraft, or the Concorde, which enabled civil supersonic flight for the first time, were still derived from earlier technical foundations. Modern commercial aircraft predominantly consist of familiar architectures and subsystems, with only partial replacement by new or adapted solutions. According to this definition, aircraft programs can be divided into two categories.

Next-generation developments introduce new system architectures or technological advances, but they are based on proven principles. For instance, the Boeing 787's environmental control system was completely converted from pneumatic to electric power,

enabling improved efficiency and better controllability [18, p.11]. The Airbus 380 replaced the classical "3H" architecture, using three independent hydraulic circuits, with a "2H2E" configuration. In this architecture, two hydraulic networks are retained but complemented by two electrically powered networks [19, p.6],[1, pp.18-19]. In addition, the A380 was the first commercial aircraft to introduce Integrated Modular Avionics (IMA) on a large scale. Instead of relying on many separate, function-specific computers, IMA's combine multiple functions on a few standardized computing modules that are linked through a high-speed data network. This modular setup reduces the amount of wiring and hardware needed, simplifies maintenance, and enables software updates without replacing the underlying hardware, which helps to extend the overall lifetime of the system [20, pp.4-16]. These projects are characterized by their ability to make technological progress without fundamentally changing the overall concept of commercial aircraft.

Conversely, *evolutionary variants* largely retain the basic architecture and focus on targeted upgrades or optimizations of specific components. These aircraft types are also commonly referred to as *derivative aircraft* in the industry, but represent evolutionary developments from a systems architecture perspective. The A320neo, for example, utilizes the proven A320 architecture and combines it with more efficient engines (LEAP-1A or PW1100) [21, p.3], redesigned wingtip winglets, and new composite materials [22, p.3]. In fact, industry reports suggest that as much as 95% of the parts are identical between the A320ceo (conventional engine option) and the A320neo (new engine option), underlining the incremental nature of such developments [23]. The Boeing 737 MAX is also an evolutionary variant. It incorporates new LEAP-1B engines, redesigned wingtip winglets, and modified front landing gear while maintaining the basic architecture [22, pp.1-2]. The primary development effort here lies in integrating new components into existing systems.

Regardless of the development approach, program durations illustrate the scale of the problem. According to a NASA report, the development time for next-generation and derivative aircraft programs differs markedly. Next-generation developments typically require around six to eight years from the initial configuration and market analysis to the first delivery, whereas derivative (i.e., evolutionary) products are usually completed within roughly 28 to 40 months [24, pp.3-6]. Although the report dates back to 2001, these figures remain broadly consistent with more recent examples. The Boeing 737 MAX, for instance, required about six years of development, which is unusually long for a derivative program and has been attributed to constrained research and development resources during early phases [25, p.115]. The Airbus A320neo, another derivative, took about four years to reach its first flight [26]. It should also be noted that there are no official data from manufacturers regarding exact development durations. Publicly available figures usually cover only the period from program launch to first flight, while the subsequent certification process often adds considerable time before entry into service. These lengthy cycles underscore the organizational and technical complexity of modern aircraft programs, highlighting the significant impact of the time- and resource-intensive analysis step on the overall duration.

1.1. Problem Statement

The initiation of a new aircraft program is usually triggered by factors such as the end of service life of existing models, shifts in market demand, technological advances, or new regulatory requirements [27, p.7]. In the early concept phase, Top-Level Aircraft Requirements (TLARs) such as passenger capacity, range, or emissions targets are defined and then refined into system-level specifications across different departments [27, p.7].

Each system is primarily developed independently within defined boundaries, while aligning with the overall aircraft architecture. This dissertation is based on the fundamental assumption that system development within such programs typically begins with an analysis of existing systems. Engineers examine which systems, subsystems, requirements, or components from previous programs can be adopted, adapted, or developed anew. While there is no public evidence that every program explicitly follows this procedure, industry standards such as ARP4754B [28, pp.69-73] prescribe early requirements definition and safety analyses that frequently draw upon prior systems. This assumption reflects a standard, though under-documented, element of early system development. The responsible engineers must review, compare, and evaluate this documentation to make informed decisions about reuse, adaptation, or new development.

The associated analysis is highly iterative and characterized by repetition. Documents are checked in multiple rounds, information from various sources is compared, and interim results are updated continuously. This process is not only labor-intensive but also prone to error because inconsistencies, duplications, and gaps in documentation can easily be overlooked.

The shortage of skilled engineers further amplifies this challenge. In Germany, for example, two out of three engineering firms reported vacancies in 2022 [29], with demographic projections suggesting shortages of 84,000 to 390,000 engineers by 2029 due to "baby boomer" retirement [30]. Declining university enrollments exacerbate this shortage: the number of mechanical engineering students (in Germany) decreased from 41,700 in 2013/14 to 22,700 in 2022/23 [31], while maintaining a dropout rate of over 30% from bachelor programs [32, p.7]. This combination of increasing demand, demographic shifts, and reduced training capacity intensifies the pressure on existing staff, leaving less time for knowledge transfer and making it increasingly difficult to evaluate and reuse existing technical knowledge systematically.

This is precisely where *Generative Artificial Intelligence* (GenAI) reveals its potential. GenAI refers to methods trained on extensive text and data sets that are capable of generating new content and providing contextually relevant answers [33, p.1, 7, 16]. Its practical relevance became evident with the launch of ChatGPT in November 2022 [34], which marked the transition of GenAI from a specialized research field to a technology accessible to millions of users worldwide [35, p.1]. User acceptance was unprecedented, with ChatGPT reaching one million users within five days after its release. For comparison, it took Instagram 2.5 months and Spotify five months to reach the same milestone [36].

Industrial adoption has accelerated equally due to the accessibility of GenAI. A recent McKinsey study from March 2025 reveals that 71% of the surveyed companies now use GenAI in at least one business function (e.g., product development, software engineering, supply chain management). This is a significant increase from approximately 33% in 2023 [37, p.15]. The widespread industrial integration highlights the technology's evolution from an experimental tool to an essential business capability in two years. The MIT Technology Review [38] recognizes GenAI as one of the ten most groundbreaking technologies of 2025. In particular, it highlights "*generative AI search*", which "*promises to make finding what you're looking for simple and quick*" by combining information from multiple sources to provide answers [38]. This suggests that the integration of GenAI into engineering development processes is not a question of if, but when and how.

GenAI is primarily based on *Large Language Models* (LLMs). LLMs are neural architectures, which have been trained on large sets of text documents and can understand, process, and generate natural language [39, p.4], [40, pp.1-2]. LLMs demonstrate remarkable language processing, reasoning, and text generation capabilities. This opens up new possibilities for systematically analyzing engineering documentation and making relevant knowledge available for development processes.

However, there are three key limitations to its application in aircraft development. First, the processing volume is limited by the context window[2], meaning that extensive document collections cannot be analyzed in a single step [41]. Second, LLMs cannot directly access domain-specific data sources such as internal or external document databases, certification records, or legacy system documentation [42, p.402]. Third, LLMs tend to hallucinate, i.e., generate content that sounds plausible but is factually incorrect [40, pp.37–38]. In areas such as the aircraft field, this is a significant risk because design decisions must be based exclusively on verifiable knowledge.

Retrieval-Augmented Generation (RAG) is a promising solution that combines the language capabilities of LLMs with the reliability of documented sources. The process works as illustrated in Figure 1.2. A user submits a query, which is transformed into a vector representation and used to search a vector database (Retrieval). Relevant information segments are retrieved and added to the original query to form an enriched prompt (Augment). Finally, the LLM processes this contextualized prompt to generate a coherent and traceable answer (Generate) [43]. In the illustrated example, the user query "*What engine options are available for the A320neo, and who are the manufacturers?*" demonstrates this workflow. The system retrieves verified data identifying two suitable engine types, the CFM International LEAP-1A and the Pratt & Whitney PW1100G-JM, and integrates them into the augmented prompt. Based on this input, the LLM generates a final answer such as: "*The A320neo can be equipped with either the CFM International LEAP-1A or the Pratt & Whitney PW1100G-JM geared turbofan engines.*"

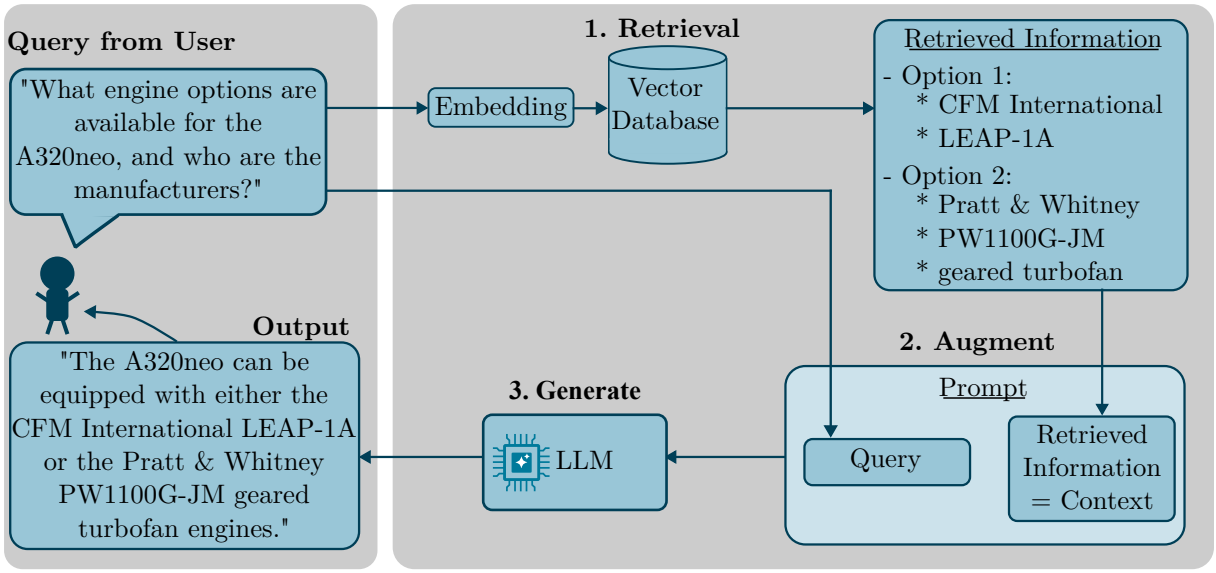

Figure 1.2.: Principle of a classical RAG system: user queries are enriched with retrieved context from structured knowledge bases before being processed by the LLM, adapted from [43].

RAG is especially useful for analyzing aircraft system documentation because it can provide engineers with faster and more reliable answers to key questions, such as which subsystems can be adopted? Which components require modifications? And which areas actually require new developments?

However, traditional RAG systems are typically designed for query-response logic, as can be seen in Figure 1.2. Each query is answered in isolation without developing a deeper

[2]The context window refers to the maximum text length that an LLM can process at one time. If this is exceeded, the prompt execution does not take place at all.

understanding of the interconnected technical relationships. This is insufficient for the systematic analysis of aircraft systems. Iterative strategies are required to bring together information from many sources and enable sound decisions regarding reuse, adaptation, or new development.

This observation leads to a key research gap. While LLMs offer language capabilities, and RAG systems provide a reliable information base, there is currently no integrated methodology that combines these approaches to assist engineers in analyzing and reusing existing technical systems systematically. Addressing this gap constitutes the central objective of the present dissertation.

1.2. Research Assumptions and Questions

The research presented in this dissertation is based on a set of explicit assumptions. First, it is assumed that accelerated analysis of existing systems represents a key lever for shortening today's development cycles. In the context of aircraft development, this means that new systems are typically not created from scratch, but rather emerge from the systematic reuse and recombination of existing architectures and subsystems. Consequently, most developments can be understood as structured combinations of proven components. Such recombinations, however, require a comprehensive analysis of existing systems to identify suitable components, understand their interactions, and derive applicable design principles. Within this dissertation, this analysis step is regarded as particularly time- and resource-intensive, since it relies on extensive, heterogeneous[3], and distributed[4] documentation. Finally, it is assumed that systematic knowledge extraction and processing can be supported and partially automated by RAG-based processes, thereby improving analysis efficiency without compromising the traceability of design decisions.

In consideration of the aforementioned context, the primary research question of this dissertation emerges: **Can a GenAI-based methodology accelerate the development of aircraft systems by systematically extracting and structuring knowledge about existing systems, determining what can be reused or adapted based on new requirements, identifying what must be newly developed, and supporting the generation of technically coherent system architectures?**

Four interrelated sub-questions derived from the central research question examine how GenAI-enhanced approaches can accelerate aircraft system development by supporting the analysis of existing systems and the development of new architectures.

- **RQ1: Knowledge Extraction Challenge**
  How can the single-query limitations of traditional RAG systems be systematically overcome to enable comprehensive engineering knowledge extraction from technical literature while maintaining complete source traceability?
  This question examines the core challenge in systematic literature analysis: conventional RAG systems process single queries in isolation, making comprehensive domain exploration time-intensive and incomplete. The focus lies on determining whether automated approaches can achieve the systematic coverage that engineers currently accomplish through months of manual analysis while preserving complete source traceability.

[3]The term *heterogeneous documentation* denotes diverse document types (e.g., specifications, design reports, standards, patents) that vary in quality, depth, reliability, terminology, and structure.

[4]In contrast, *distributed documentation* refers to information fragmented across organizational units, projects, or storage systems, lacking a unified structure or centralized access, which complicates systematic retrieval and comparison.

- **RQ2: Knowledge Integration Challenge**
  Is it possible to systematically synthesize fragmented technical information into a coherent engineering document that meets established systems engineering standards?
  This question examines whether LLMs can effectively transform dispersed technical knowledge into a structured engineering document. This question addresses the critical capability of consolidating question-answer pairs from the RAG-based process, with information regarding component specifications or design principles, into coherent technical reports that maintain engineering standards while enabling informed decision-making for subsequent system design phases.

- **RQ3: Implementation Feasibility**
  What are the critical configuration parameters for deploying a GenAI-based methodology in aircraft engineering environments?
  This question addresses the configuration, including optimal retrieval strategies, model selection criteria, prompt engineering approaches, and knowledge base composition that achieve performance levels suitable for safety-critical applications while maintaining computational efficiency.

- **RQ4: GenAI-Generated Architecture Via Systematic Recombination**
  Can a GenAI-based methodology systematically combine knowledge from existing systems with new requirements to generate technically coherent system architectures?
  This question is evaluated in two complementary dimensions:
  (i) *Documentation & Reuse/Adapt/New: Can the methodology, based on the synthesized document, systematically assess which elements are reusable, which require adaptation, and which must be newly developed?*
  (ii) *Architecture Generation: Do these structured foundations enable the creation of new yet technically coherent architectures, and should these be regarded as robust solutions or primarily as initial drafts that require human optimization?*
  These questions evaluate whether the methodology can leverage comprehensive knowledge from existing systems to address engineering challenges through systematic recombination rather than entirely new invention. The investigation examines the boundary between GenAI-generated architectural concepts and human-required design optimization, establishing the optimal division of responsibilities in human-GenAI collaborative engineering processes.

To address these research questions, this dissertation develops a comprehensive 4-Phase methodology, called SAGE, that systematically transforms separated pieces of technical literature into system architectures. SAGE stands for *System Architecture through Generative Engineering* and the methodology consists of the following phases:

- Phase 1 - Data Preprocessing: creates a reliable knowledge foundation by transforming large volumes of technical documentation into a structured and searchable format. This step ensures that information can be retrieved efficiently while maintaining its original context and meaning.

- Phase 2 - Extended RAG with Recursive Knowledge Extraction: enhances traditional retrieval processes by enabling systematic exploration of technical domains rather than answering isolated queries. It ensures that interconnected topics are covered comprehensively and that every generated result remains fully traceable to its original sources.

- Phase 3 - LLM Knowledge Synthesis: consolidates fragmented information from Phase 2 into a coherent engineering document that aligns with established systems engineering frameworks. This step transforms dispersed details into structured reports that support sound engineering decision-making.

- Phase 4 - LLM Architecture Generation: applies reasoning LLMs to propose system architectures that address new requirements. It ensures that generated concepts remain technically consistent and grounded in established engineering principles.

The SAGE methodology is developed under the constraint that no access to proprietary documentation from aircraft manufacturers was available. Consequently, all phases of the methodology are implemented and tested on publicly accessible sources, including scientific literature and patents. This has two important implications. First, the evaluation does not aim for a one-to-one replication of historical program baselines. Instead, the methodology is evaluated through a three-step procedure: (i) consolidating the description of a conventional reference system from public evidence and (ii) evaluating reuse, adaptation, or new development when requirements are changed, and (iii) testing whether technically coherent system architectures can be generated through systematic recombination of existing knowledge.

To demonstrate this process in practice, two case studies were selected: the *Environmental Control System (ECS)* and the *Thermal Management System (TMS)* for fuel cell integration. In the ECS case study, the methodology is first applied to consolidate knowledge about the conventional pneumatic ECS architecture and then evaluated against the requirements of the electrical ECS used on the Boeing 787. The TMS case study, in contrast, illustrates the methodology's ability to address emerging domains where integration challenges and novel technologies play a central role. This evaluation approach ensures that SAGE is assessed under realistic constraints while demonstrating its capability to distinguish systematically between reuse, adaptation, and new development decisions even without access to proprietary design archives.

1.3. Thesis Structure and Reading Guide

This dissertation is organized into eight chapters, as can be seen in Figure 1.3. Together, these chapters establish the theoretical foundations, develop methodological contributions, evaluate the approach through practical applications, and synthesize findings for future research directions. The structure follows a logical progression from problem identification through solution development to comprehensive evaluation and analysis.

Foundation Chapters (1 - 2)
The research begins with establishing the theoretical and contextual foundations, which are necessary for the understanding of the intersection of GenAI technologies and aircraft systems engineering. This chapter provides a comprehensive introduction that contextualizes the research within current industry challenges, articulates the specific problem statement, and defines the research objectives and methodology. This introductory chapter establishes the motivation for a GenAI-based methodology while describing the scope and expected contributions of the research. Chapter 2 presents a systematic state-of-the-art analysis that examines current approaches across multiple domains relevant to this research. It provides a critical analysis of document-based systems engineering approaches and their limitations, explores the evolution and implementation challenges of model-based systems engineering, investigates the specific context of civil aircraft development as a knowledge-intensive domain, and establishes the theoretical foundation of GenAI with particular emphasis on RAG technologies. This comprehensive literature review identifies specific research gaps and establishes the scientific foundation for the methodology development, see Chapter 3.

Methodology Development (3 - 5)
The core methodological contributions are presented across three interconnected chapters that systematically develop the GenAI-based methodology. Chapter 3 introduces the

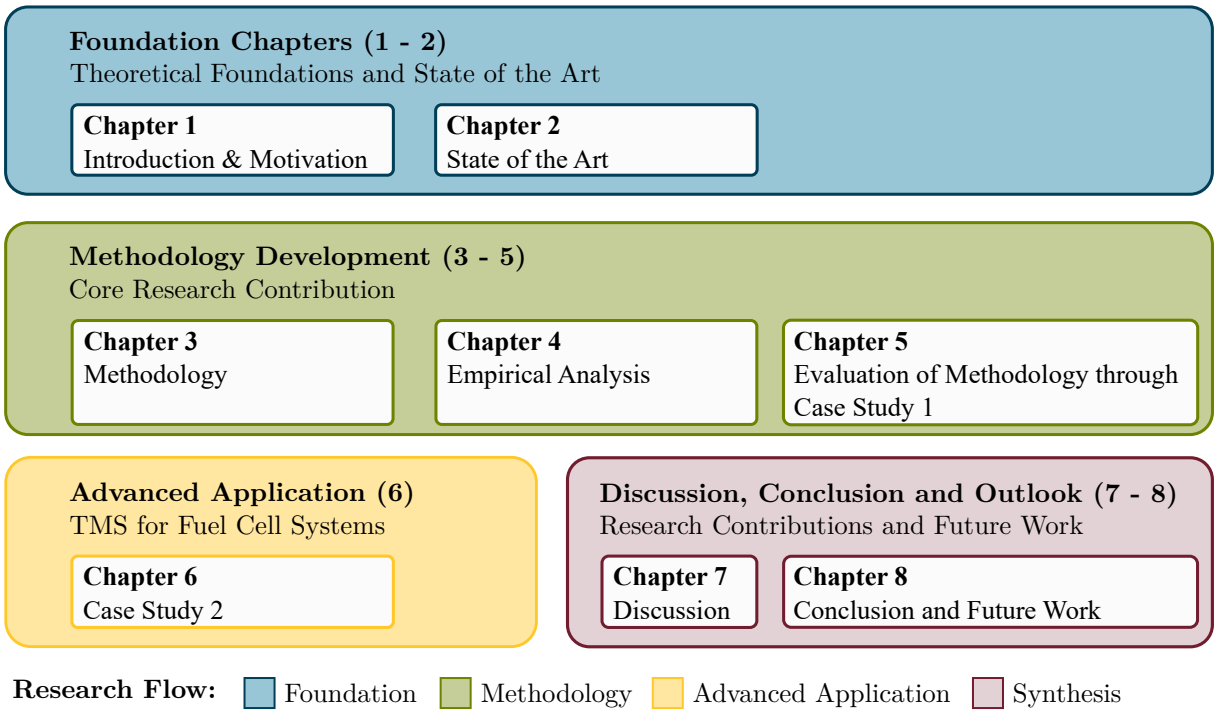

Figure 1.3.: Structure of the dissertation

developed methodology, which establishes the data preprocessing process, the recursive knowledge acquisition method, and the methods for knowledge synthesis and architecture generation. This provides the conceptual foundation and initial implementation of the proposed methodology. Chapter 4 extends the methodology through an empirical analysis of the configuration. Through empirical studies including vector store optimization, chunk selection analysis, LLM model comparison, and prompt engineering optimization, this chapter provides practical implementation guidance. Chapter 5 demonstrates the practical application of the developed methodology through a comprehensive evaluation using the ECS as a case study. This chapter provides detailed implementation guidance, evaluates the quality and reliability of generated knowledge synthesis and architectures, and establishes an evaluation for assessing GenAI-generated system solutions against established technical and design criteria.

Advanced Application (6)
Chapter 6 extends the methodology application to the TMS for fuel cell integration, representing a different application domain. This chapter demonstrates the methodology's capability to address emerging technologies and complicated system integration challenges while providing a comprehensive evaluation of both technical performance and design innovation potential.

Discussion, Conclusion, and Future Directions (7 and 8)
Chapter 7 synthesizes the findings across the methodological and case study chapters, evaluates iteration time and workload, and positions the SAGE methodology within existing development approaches. It also discusses advantages and limitations, as well as the expected impact on industrial development processes. The dissertation concludes with Chapter 8, which provides the overall contributions and closes with an outlook on future research directions.

Reading Recommendations
This dissertation addresses diverse audiences with varying interests and expertise levels:

- Systems Engineers should focus on Chapters 3, 5, and 6 for a practical understanding of methodology and implementation guidance. These chapters offer concrete approaches for integrating GenAI technologies into existing systems engineering workflows.
- AI Researchers will find particular value in Chapters 4 and 5, which provide detailed technical optimization analysis and performance evaluation of the developed methodology applied to technical domains. These chapters offer insights into parameter optimization, model selection, and domain-specific adaptation strategies.
- Aircraft Practitioners should emphasize Chapters 5 and 6, which provide domain-specific applications and industry-relevant case studies. These chapters demonstrate practical implementation considerations and give a realistic assessment of current technology adoption potential within aircraft development environments.
- Academic Researchers are recommended to read the complete dissertation, with particular attention to Chapters 3-5, which establish the theoretical foundations and systematic approaches that can be adapted to other engineering domains and research contexts.

The dissertation structure enables readers to engage with content at different levels of detail while maintaining logical progression from theoretical foundations through practical implementation to comprehensive evaluation and future research directions. To ensure terminological clarity, this dissertation applies central terms such as *methodology, method, process, framework, approach*, and *model* in a consistent manner. Their definitions and distinctions are summarized in Appendix A.1 on page 177, along with other key terms such as *system* and *engineering*, which form the basis of the terminology throughout this dissertation.

1.4. Summary

Bill Gates aptly summarized the transformative potential of GenAI: "*Generative AI has the potential to change the world in ways that we can't even imagine. It has the power to create new ideas, products, and services that will make our lives easier, more productive, and more creative.*" [42, p.15]

For aircraft system development this potential translates into faster knowledge extraction, more systematic analysis of existing architectures, and more reliable foundations for design decisions. This dissertation, therefore, develops the SAGE methodology, which systematizes and accelerates the previously manual, time-, and resource-intensive processing of existing aircraft system documentation. The methodology consolidates and structures existing systems along RFLP logic, evaluates them against new requirements to determine which elements can be reused, which require adaptation, and which must be newly developed, and uses this structured knowledge base to generate consistent conceptual system architectures as starting points for further engineering optimization.

2. State of the Art

The transformative potential of GenAI, as discussed in Chapter 1, indicates that it could also play a significant role in the future development of aircraft systems. To assess this potential and to position the proposed SAGE methodology within the broader research landscape, it is necessary to understand both traditional systems engineering methodologies and recent developments in AI technologies.

This chapter, therefore, examines three perspectives:

- Classical document- and model-based engineering methodologies,
- The knowledge-intensive characteristics of civil aircraft programs, and
- AI-supported methods, including recent advances in large language models and retrieval-augmented generation.

Taken together, these perspectives reveal the limitations of current approaches and define the requirements that guide the methodology presented in Chapter 3.

Systems Engineering (SE) emerged in the late 1950s as a transdisciplinary discipline to manage the growing complexity of developing technical systems [44, p.13], [45, p.18], particularly in large-scale aircraft and defense projects in the United States [46, pp.55-56]. The exponential growth in technological complicatedness,[1] from simple mechanical systems to today's cyber-physical systems involving millions of interactions, has since driven the continuous evolution of SE methodologies. The following section examines the well-established SE methodology and identifies the knowledge management constraints that motivate continued advancement.

2.1. Document-Based Systems Engineering

SE, as defined by the International Council on Systems Engineering (INCOSE), is a transdisciplinary approach aimed at managing the increasing complicatedness of multidisciplinary technical systems and minimizing their risks [12, p.11], [46, p.43], [47, p.45], [48, p.386]. This systematic methodology enables structured development through requirements documentation, functional analysis, and architectural modeling across all system levels [46, p.43]. According to the National Aeronautics and Space Administration (NASA) [49, p.3], this discipline aims to develop functional and safe systems that can fulfill requirements despite often conflicting interests and multiple or partially competing constraints. This integrative methodology promotes collaboration among engineers from various disciplines to develop cross-disciplinary solutions [49, p.3].

Several publications have made significant contributions to the development of SE. These include Arthur D. Hall's publication "*A Methodology of Systems Engineering*" in 1962 that established the theoretical foundation. INCOSE's "*Systems Engineering Handbook*

[1] In this dissertation, the noun *complicatedness* is used to denote the state of being *complicated*. While the term *complexity* is more common in English, it is here reserved exclusively for systems with complex properties.

V1.0" introduced in 1994 a standardized methodology that has since been continuously refined [12]. The introduction of an international SE standard occurred in 2002 with ISO/IEC/IEEE 15288:2002 [11], which established a unified understanding for the first time and facilitated legal integration. This standard has been continuously revised and republished. In 2012, the "*Systems Engineering Body of Knowledge*" (SEBoK) [46] was published, reflecting the SE community's efforts towards a unified and current knowledge representation, mainly corresponding to the INCOSE handbook.

Although SE provides the essential structure for managing system development, the practical implementation of these principles requires systematic process models to guide engineers through the development lifecycle. In this dissertation, a *model* is defined following Eigner et al. [47, p.80] definition as a "*representation or template for a system or process. A model can be a conceptual (e.g., mathematical, information technology) or a physical (e.g., scaled material prototype) structure*" (author's translation)[2] [47, p.80]. Following this definition, a *process model* describes an abstract representation of the structured sequence of activities, phases, and decision points that organize and guide engineering work toward defined objectives. Such models provide a detailed and practice-oriented framework that can be adapted to specific technical and organizational needs. They serve as a standard reference for developers, facilitate communication within teams, and support the systematic monitoring of development progress [50, p.213].

Various engineering disciplines have developed specialized process models tailored to their specific requirements. For example, software development employs iterative process models such as the Waterfall, Agile, and Spiral models. Detailed descriptions of these and other process models across engineering domains are provided at Eigner et al. [47, pp.15-48]. Among these process models, the V-model has emerged as the predominant one for systematically organizing SE activities, particularly in mechanical and aircraft engineering.

2.1.1. The V-Model

The V-model illustrates the phases of system development through a series of closely interconnected steps, from defining requirements to designing, implementing, verifying, and validating the system. Initially introduced by Boehm in 1979 for the development of software and later refined for SE [51, p.344], the V-model represents the decomposition of systems into manageable elements on the left branch, followed by the step-by-step integration of these elements into the overall system on the right branch, as can be seen in Figure 2.1 [52, p.1].

As illustrated in Figure 2.1, *system development* encompasses the entire V-shaped process, including both the decomposition activities on the left branch and the integration activities on the right branch, along with verification and validation processes. While the term *system development* is often used as a generic descriptor for the entire V-process, this dissertation explicitly distinguishes it from *system design* to ensure conceptual precision. *System design* refers specifically to the decomposition phase represented by the left branch of the V-model [53, p.7], in which stakeholder needs are systematically transformed into concrete system specifications through requirements analysis, functional decomposition, and the development of logical and physical architectures.

[2]Original in German: "*Ein Modell ist ein Abbild bzw. ein Vorbild für ein System oder einen Prozess. Ein Modell kann ein begriffliches (z.B. mathematisches, informationstechnisches) oder ein physisches (z.B. maßstäblicher stofflicher Prototyp) Gebilde sein.*" [47, p.80]

The hierarchical approach considers the operational context of the System of Interest (SOI) and its integration within the overarching System of Systems (SoS) [2, p.4]. Complicatedness is managed through systematic subdivision, with subsequent decomposition activities focusing on defining the solution. When a solution element remains too complicated, it is treated as a subsystem, which is then subdivided into subordinate elements.

The V-model is structured around the RFLP framework,[3] which stands for Requirements, Functions, Logical Architecture, and Physical Architecture. This concept provides a hierarchical framework for system specification:

1. Requirements: System engineers subdivide between functional requirements that describe what the system shall do and non-functional ones that determine quality characteristics such as performance, reliability, and safety [47, p.57].
2. Functions: Tasks and capabilities that the system must provide to meet defined requirements. These elements represent a detailed specification of the system's operational behavior [47, p.85].
3. Logical Architecture: An abstract description of the system structure. It defines the logical elements and their interactions, and determines how the system's functions should be distributed across its logical elements, without specifying the physical implementation [47, p.86].
4. Physical Architecture: Concrete system implementation describing the actual physical distribution of hardware components, networks, and resources required for operation [47, p.86].

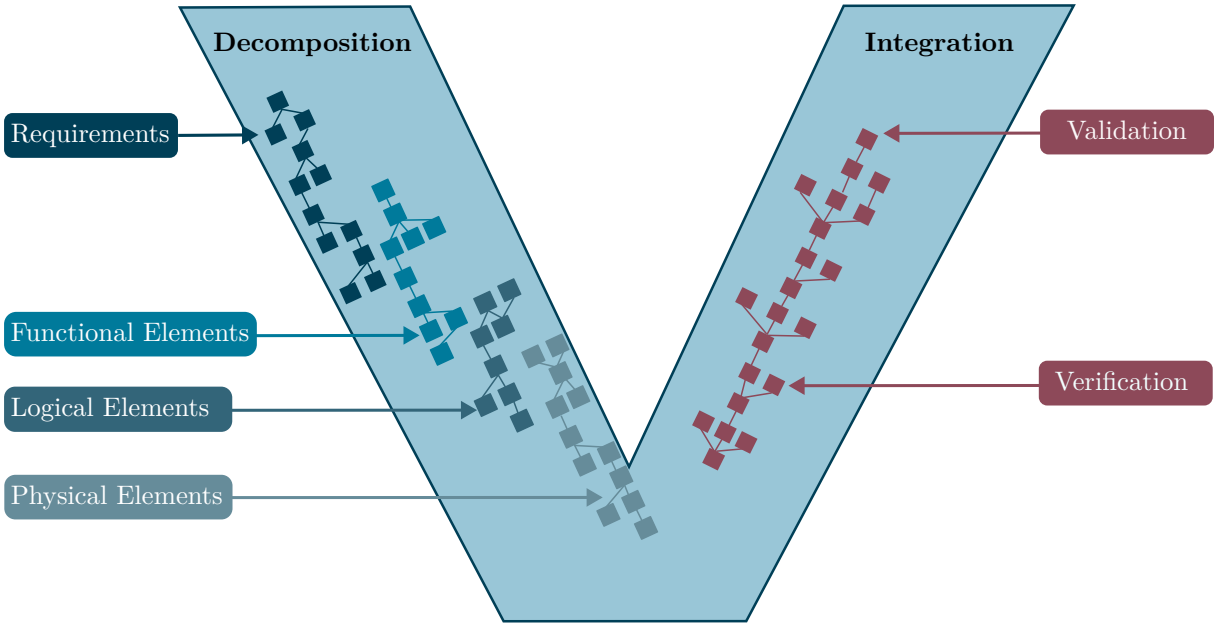

Figure 2.1.: RFLP framework in the V-model, adapted from [44, p.158]

These four types of elements provide a specification of the system throughout all phases of design. They enable a systematic definition of the design solution, starting from abstract requirements, progressing through functional decomposition, and culminating in a

[3]The term *framework* is used here because the RFLP concept provides a structured scheme that defines the logical relationships between requirements, functions, logical elements, and physical components. Rather than a prescriptive process, it serves as a methodological backbone, guiding the structuring and traceability of system information throughout development.

concrete physical implementation. The V-model approach provides structure and traceability for the development of complicated systems, forming the backbone of modern systems engineering practices.

Nevertheless, while the V-model and RFLP framework provide a systematic structure for development processes, their document-centric nature creates challenges that become critical limitations as systems become more complicated. The complicatedness of managing interdisciplinary knowledge expands. Additionally, the V-model's generic nature requires extensive adaptation to specific application domains, a process commonly referred to as *tailoring*. In SE practice, tailoring denotes the project-specific adjustment of processes, documentation, and activities to the given organizational and regulatory context. It also covers the definition of suitable review and decision-making procedures that comply with applicable legal and certification requirements [12, pp.162-165, 265], [44, pp.34-36]. For instance, while developing a simple sensor might require only a reduced set of documentation and verification steps, designing an aircraft system demands the full lifecycle of activities, additional safety analyses, and domain-specific certification processes.

2.1.2. Challenges of Document-based Systems Engineering

Document-based Systems Engineering (DBSE) uses systematic processes to capture, manage, and communicate information related to system requirements and development activities. During development, numerous critical documents are created to guide, structure, and track project progress [54, p.106]. These documents serve as a knowledge repository.

While this approach provides the backbone of many industrial projects, its document-centric nature also introduces recurring limitations. The following list summarizes those limitations:

- Scale and system complicatedness: As systems have become more complicated, document-centric coordination reaches practical limits [54, p.101], [44, p.68].
- Change propagation, consistency, and traceability: Updates must be replicated across many documents, and maintaining bidirectional traceability (requirements, functions, architectures, verification) is effortful and often incomplete [51, pp.32-33], [12, p.189].
- Communication and cross-domain synchronization: Long textual specifications hinder a shared understanding of system behavior, interfaces, and work packages. Therefore, a coherent, system-wide view across disciplines is challenging to achieve [51, p.32], [55, p.6], [56, p.1], [57, p.2], [58, p.5], [59, p.73], [44, p.69], [60, p.32].
- Reuse, variability, and change management: Knowledge is spread out across various specifications and documents, which makes it difficult to reuse information systematically and manage variants. Consequently, it is challenging to evaluate the impact of changes [47, pp.81-82], [60, p.32], [61, p.12].
- Knowledge capture and centralization: Capturing expert knowledge in a standardized, accessible form is difficult [58, p.5], [62, p.13f], [59, p.73]. Results from different disciplines remain fragmented rather than consolidated [55, p.6].

Taken together, these limitations motivated the development of Model-based Systems Engineering (MBSE). MBSE shifts the focus from documents to formalized system models in order to improve consistency, traceability, and cross-domain integration. It aims to minimize the number of documents traditionally generated in the development process and, at the same time, reduce the effort required to maintain consistency through a central system model [60, pp.32-33], [47, p.81].

2.2. Model-Based Systems Engineering

INCOSE defines MBSE as "*the formalized application of modeling to support system requirements, design, analysis, verification, and validation activities beginning in the conceptual design phase and continuing throughout development and later life cycle phases*" [46, p.202]. This section examines the theoretical foundation, core principles, advantages, and remaining limitations of MBSE.

2.2.1. Theoretical Foundation and Core Principles

MBSE represents a paradigm shift in system development, moving from a document-centric way to a model-centric one. In this approach, integrated digital system models serve as the primary means of capturing, analyzing, and communicating information [44, p.53]. Unlike the document-based practice, which stores information in isolated documents, MBSE maintains all system knowledge within semantically connected model structures, ensuring consistency and enabling systematic analysis.

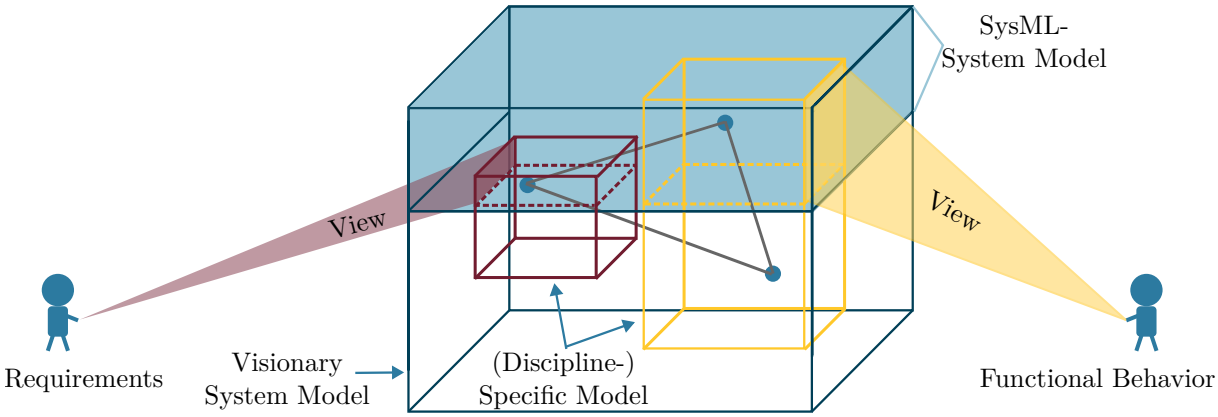

Figure 2.2.: Visionary system model vs. SysML system model, adapted from [44, p.55]

According to Gräßler and Oleff [44, p.53], the system model serves as the core of model-based systems development. Building on the general model definition established earlier, a *system model* represents a formal, structured representation of a technical system. This model can capture various aspects of the system, depending on the specific domain and purpose [44, p.53]. In the context of SE, these aspects typically include requirements, functions, logical elements, and physical components, though other engineering disciplines may emphasize different system characteristics.

The visionary goal is to consolidate all relevant system-specific information within a unified model. As illustrated schematically in Figure 2.2, this visionary system model encompasses various discipline-specific models [63, p.2], [47, p.82], [44, p.53]. Each describes a specific aspect of the system under development, such as behavior or structure. Ideally, the model would serve as a "single source of truth" for all system-relevant information. However, according to Gräßler and Oleff [44], this vision remains practically unattainable due to the enormous complicatedness and administrative overhead associated with such comprehensive models. Instead, current practice employs a combination of overarching system models and discipline-specific, detailed models, aiming to eliminate silos and maintain semantic relationships across all elements [47, p.82].

Successful MBSE implementation requires the coordinated integration of three essential components, language, tools, and methodology, that form an inseparable methodological foundation [64, p.29]. The *modeling language* defines how systems are formally represented. The Systems Modeling Language (SysML), introduced in 2007 as an extension of UML, is the dominant choice. It offers different diagram types to examine systems from various perspectives [64, pp.29–32], as can be seen in Figure 2.3. Structural diagrams illustrate how a system is constructed and which components are interconnected. Behavior diagrams illustrate what the system does: they capture processes, use cases, and state changes. A significant innovation of SysML v1 is the requirement diagram, which enables systematic modeling of requirements and ensures their traceability throughout the entire development process for the first time. Although SysML v1 has become the standard language, its practical use has revealed limitations in interoperability, scalability, and integration with modern digital engineering tools [65, p.1], [66]. These challenges have led to the development of a new version of the language. SysML v2 was released on July 21, 2025 [67], and is expected to address many of the current weaknesses through fundamental improvements: enhanced portability between different modeling tools, an open Application Programming Interface (API) that enables integration into heterogeneous model ecosystems, and text-based input formats that make the language accessible to GenAI sources for automated model creation [68, p.2], [69, pp.2–5, 15]. These advances promise to make SysML significantly more flexible and interoperable in modern system development environments.

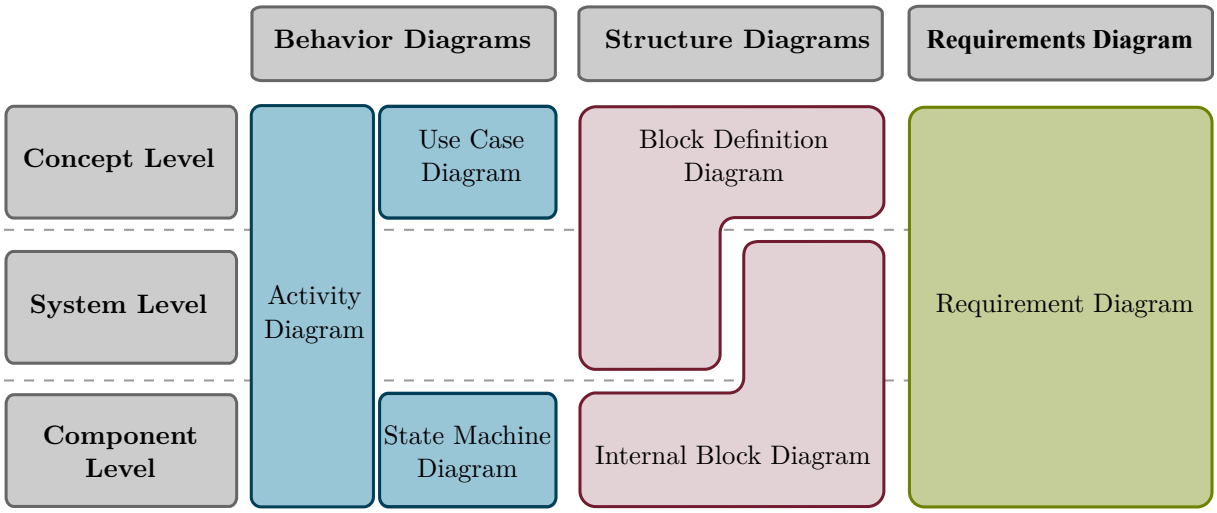

Figure 2.3.: Overview of existing diagram types in SysML v1, adapted from [58, p.13]

Modeling tools are the software platforms for creating and managing models. Solutions such as Cameo Systems Modeler, Enterprise Architect, Rational Rhapsody, and Capella ensure syntactic correctness and model consistency. Tool selection significantly influences implementation success, as factors such as language support, interoperability, performance features, and integration with existing development environments directly affect user adoption and productivity outcomes [64, p.30].

Methodologies show how languages and tools are applied in concrete development projects. Well-known methodologies include OOSEM, IBM Harmony SE, the Vitech MBSE methodology, JPL State Analysis, and Dori Object-Process Methodology [12, p.190]. Despite different names, all established methodologies follow a similar basic pattern that corresponds to the V-model. The typical workflow includes:

1. Stakeholder requirements: The process begins by capturing stakeholder requirements, encompassing expectations from various interest groups, individuals, and

regulatory frameworks. Subsequently, these high-level stakeholder needs are refined into specific system requirements that precisely articulate what the technical system must accomplish.

2. Modeling system context: Many methodologies capture the environment in which the system operates and the external factors with which it interacts. This context helps to identify possible external influences early.
3. Defining functions: Engineers identify which tasks the system should fulfill and derive functional requirements from them [70, p.183]. These are then implemented as specific system functions.
4. Developing architectures: Deriving architectures from functional and non-functional requirements encompasses modeling the system structure. This includes both the *logical architecture*, which defines the system's technical concepts and principles, and the *physical architecture*, which describes the system's concrete components and their interrelationships.
5. Implementing, analyzing, and integrating: The defined components are implemented at the hardware and software levels, tested through simulations or testing, and finally assembled into the overall system.

These commonalities demonstrate that MBSE approaches rely on proven systems engineering principles, despite differences in terminology and emphasis. The underlying development logic remains remarkably consistent across various methodologies.

2.2.2. Advantages of MBSE

MBSE attempts to overcome the fundamental weaknesses of document-based approaches by replacing fragmented documents with integrated models. The most evident advantage is that changes to model elements are automatically propagated throughout the entire model. This automation eliminates the tedious manual synchronization of various documents, and the typical contradictions between specifications become a thing of the past [12, p.189], [60, p.33]. Engineers can systematically analyze the effects of a change on the overall system, something that is practically impossible with traditional documents.

Another significant benefit is better communication. Models are more intuitive to understand than pages-long specifications. They enable different disciplines to quickly gain a common understanding of system behavior and interfaces, thereby improving coordination across development activities [51, p.32], [55, p.6], [56, p.1]. In addition, the shared visual representations make implicit assumptions explicit, helping to identify misconceptions and prevent miscommunications early in the development process. This improved visualization significantly facilitates information exchange between stakeholders [58, p.5]. As a result, models make even highly complicated systems more manageable by providing a transparent, shared representation that aligns different disciplinary perspectives and supports coordinated development [58, p.5], [60, pp.32-33].

MBSE also supports centralized knowledge management by capturing stakeholder input and engineering knowledge in a standardized, accessible format. Results from different disciplines are consolidated within the model, creating a single structured repository that not only supports current developments but also facilitates reuse in future projects [58, p.5].

Furthermore, MBSE provides comprehensive traceability by establishing systematic connections between all system elements, from stakeholder needs through functional requirements to physical implementation and verification evidence [51, p.32f]. This seamless traceability supports systematic verification and validation [71, p.1].

2.2.3. Implementation Barriers

Despite its theoretical advantages, MBSE's practical implementation also reveals some limitations. While MBSE successfully addresses documentation problems through improved representation and consistency, this dissertation identifies four interconnected barriers that currently prevent MBSE from realizing its full potential:

- Organizational Adaptation Challenges
- Technical Implementation Limitations
- Single Source of Truth Challenge
- Persistent Manual Knowledge Acquisition

The first barrier concerns organizational adaptation challenges. The shift from document-based to model-based workflows represents a profound organizational change. Engineers with decades of experience have to adopt entirely new working methods, which demands significant learning effort and often leads to a temporary decline in productivity [71, pp.7-10]. MBSE also requires specialized knowledge of modeling languages and dedicated tools. The considerable initial investment is often difficult to justify for companies, as the expected long-term benefits remain uncertain and cannot be reliably quantified at early implementation stages [72, p.12], [71, p.7].

The second barrier involves technical implementation challenges, which frequently reinforce the organizational difficulties. Complex projects typically rely on multiple specialized tools for requirements management, simulation, and analysis. However, limited interoperability between current MBSE platforms and existing tool environments leads to inefficient data exchange and fragmented workflows [73, p.1], [74, p.3], [72, p.13]. Large-scale MBSE models, often comprising hundreds of diagrams and thousands of elements, also become difficult to manage. Practitioners report that model navigation and comprehension deteriorate as scale increases [74, p.3]. Inconsistencies across views and tools further impede collaboration: Chami et al. [74, p.1] note that "*views are not compatible and are completely disintegrated*", while Förster et al. [75, p.2535] highlight the proliferation of redundant model copies across tool landscapes. Furthermore, the variety of MBSE approaches and the lack of implementation guidance complicate standardization efforts [76, p.145]. As a result, many companies introduce MBSE gradually and in parallel with existing document-based processes. Major aircraft manufacturers exemplify this dual approach: Airbus launched its "*Digital Design Manufacturing & Services*" (DDMS) program while retaining traditional documentation workflows, and Boeing developed enterprise architecture models for over 1,000 distributed users alongside legacy processes [77, p.9].

The third barrier arises from the single source of truth challenge, where a conceptual challenge undermines MBSE's core premise of unified modeling [44, p.53]. While the methodology aims to establish a single authoritative model, SysML v1 lacks the representational richness required for actual design work and creative processes. Engineers must therefore rely on additional specialized tools and representations for conceptual design and detailed analysis, which then require translation back to SysML for formal documentation. This necessity creates a challenge: the pursuit of a single source of truth multiplies the number of models and representations, each serving different purposes in the development process. The resulting fragmentation contradicts the fundamental goal of MBSE, which is to create a uniform system representation. It also creates new challenges in terms of consistency across multiple modeling domains.

The fourth and most fundamental barrier in the context of this dissertation centers on the persistent need for manual knowledge acquisition. Despite technological advances, MBSE models still must be created and maintained manually. Engineers have to translate their

knowledge explicitly into diagrams and specifications. This process is time-consuming and error-prone. While MBSE improves consistency and traceability, it cannot automatically derive new design solutions from existing knowledge or propose new system architectures. The creative design process remains entirely dependent on human expertise. Zhang et al. [78, p.3] characterize this limitation as "*low automation and intelligence in MBSE toolchains*", noting that although existing tools support basic modeling and simulation, their overall automation levels are insufficient. Engineers have to spend substantial effort operating tools, which detracts from actual focus on system design [78, p.3].

In summary, MBSE extends and strengthens traditional systems engineering practices by improving representation, communication, and traceability. Civil aircraft development broadly follows these principles, combining both document- and model-based approaches. However, within this methodological environment, engineers repeatedly face the challenge of determining whether existing systems, subsystems, or components can be reused, require adaptation, or must be newly developed. Evidence suggests that such analyses are indeed conducted in practice, as industry standards such as ARP4754B [28] explicitly include provisions for the reuse of systems and components. Yet, this step is not formally defined within DBSE or MBSE methodologies and is therefore typically carried out in a pragmatic and resource-intensive manner.

To better understand how these methodological challenges manifest in practice, the following section examines the development of civil aircraft and their systems as a representative application domain. This contextual perspective provides the foundation for identifying the specific process constraints and knowledge-management requirements that any GenAI-augmented approach must address.

2.3. Development of Civil Aircraft and their Systems

The aircraft industry is arguably one of the most demanding application domains for document- and model-based systems engineering approaches. The development of aircraft highlights the challenges posed by technical systems, which must combine stringent safety requirements, multidisciplinary interactions, and extensive documentation. Understanding the specific procedures, regulatory frameworks, and methodological approaches of civil aircraft development is important for identifying where emerging AI technologies could solve long-standing knowledge processing issues.

2.3.1. Aircraft Development Procedure

Due to the complicated nature of aircraft design, which depends on numerous factors, there is no uniform approach to the process [79, p.7]. At Airbus, the development is structured into five phases comprising a total of 14 milestones: a feasibility study (M0-M3), a concept phase (M3-M5), a definition phase (M5-M7), a development phase (M7-M13), and a series production phase (M13-M14) [80, pp.26-28], as can be seen in Figure 2.4.

Every new aircraft development begins with an understanding of market requirements and an assessment of whether existing aircraft types can meet the demand [27, p.7]. Design processes are initiated in the feasibility phase, when factors such as the end of the service life of predecessor models, changes in legislative constraints, technological advances, or market shifts create development opportunities. Following an extensive market analysis, the Top-Level (Aircraft) Requirements (often abbreviated as TLRs or TLARs in the literature) are defined. In this dissertation, the term TLARs is used consistently. These requirements include passenger capacity, operational range, runway

constraints, and regulatory compliance targets. These TLARs directly determine aircraft functions and influence the overall architecture [81, p.12, 31], [27, p.7].

During the concept phase, the selected baseline configuration undergoes iterative refinement across multiple departments, including those dealing with airframe and system design, organized according to the Air Transport Association (ATA[4]) chapters. The aim is to achieve global optimization while considering space limitations, weight reduction, fuel consumption, and lifecycle costs [83, p.7]. Parametric studies, sensitivity analyses, and trade-off studies address variable parameters and uncertain design elements. Estimates suggest that a substantial portion of the aircraft's lifecycle costs, often cited as up to 85%, is determined during this phase. In contrast, only about 10% is defined during detailed design [84, p.2].

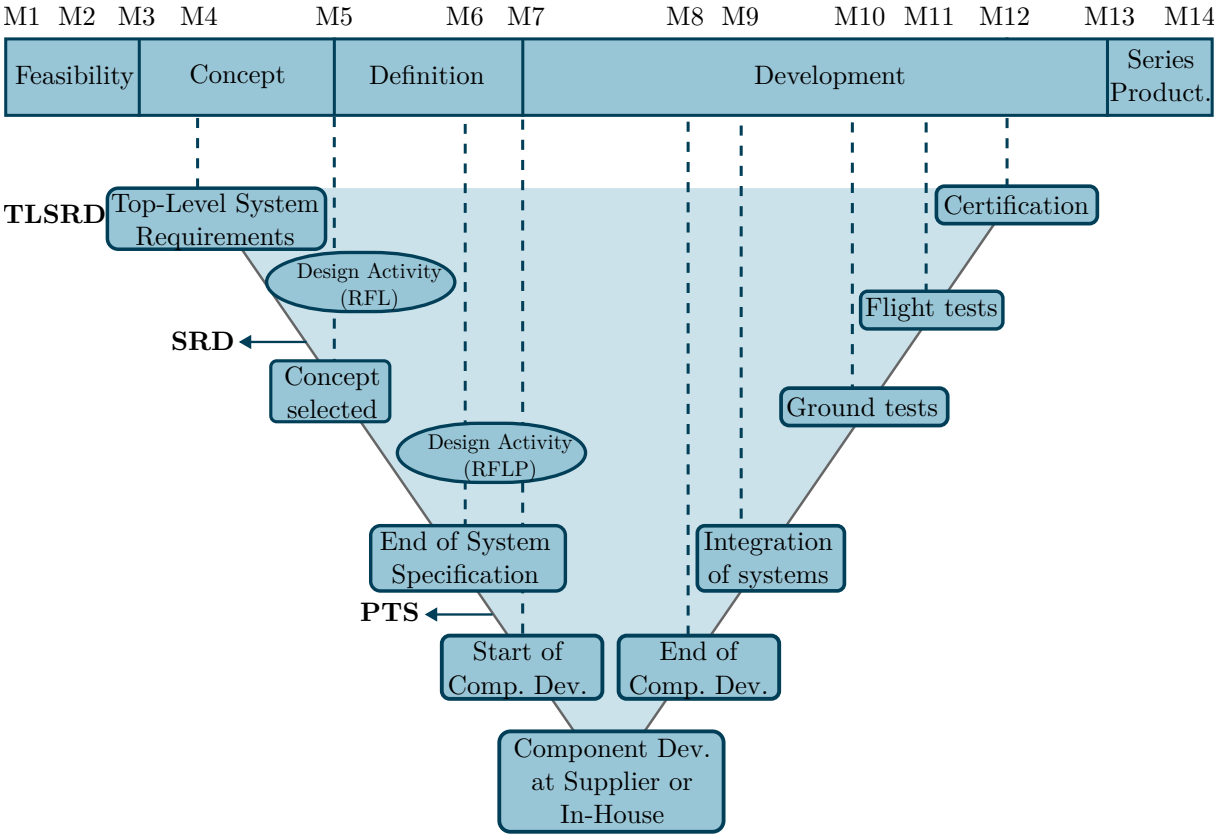

Figure 2.4.: Airbus' V-model for aircraft development, adapted from [80, p.27], [85, p.7]

The definition phase involves further configuration refinement, specifying individual systems and assemblies down to the component level while determining supplier responsibilities and manufacturing processes [27, p.13]. This phase covers the preparation of technical drawings, the creation of manufacturing standards, the production of components, and the functional testing of mock-ups, culminating in the first flight and certification [80, p.27].

[4]The ATA chapters provide a standardized numbering system that organizes aircraft systems and components into logical categories, enabling consistent documentation and maintenance procedures across the aircraft industry [82, p.138-141], [83, p.8]

2.3.2. Aircraft System Development Procedure

The development of aircraft systems also follows the V-model. The system is systematically decomposed into smaller subsystems, down to the component level, along the left branch [80, p.13]. This process is guided by the aircraft-specific guidelines ARP 4754B [28] and ARP 4761A [86], referenced in the certification specifications CS-25 [87]. These guidelines establish comprehensive frameworks for system development and safety assessment, ensuring airworthiness through rigorous documentation and analysis.

As shown in Figure 2.4, the system development process begins with the delivery of the Top-Level System Requirements Document (TLSRD), from which aircraft functions are identified and allocated to various systems through an iterative evaluation. The requirements specification culminates in the System Requirements Document (SRD), which captures functional, safety, interface, environmental, physical, maintainability, and reliability requirements. Based on the architecture definition, component requirements are documented in the Purchaser Technical Specifications (PTS) [80, p.27]. If a supplier develops the system, the PTS serves as the contractual specification for the external partner. Otherwise, it serves as the internal reference document for in-house system development.

The certification process requires comprehensive evidence to demonstrate the assignment of appropriate safety requirements, compliance with the design and its implementation, adherence to proper design processes, and satisfaction of environmental and performance requirements [88, p.295]. A detailed description of the two ARP guidelines and safety analyses can be found in Appendix A.2 on page 182.

2.3.3. Aircraft Development as a Knowledge-Intensive Domain

Aircraft development programs typically span six to eight years and cost billions of dollars, with overruns common, as seen in the Boeing 787 and Airbus 380 programs. Their scale and complexity make them exemplary cases for understanding why knowledge processing is a central challenge. Three interrelated characteristics define this domain:

- Hierarchical Complexity,
- Multidisciplinary Interactions, and
- Documentation Intensity.

First, the hierarchical, multi-level structure of an aircraft and its development necessitates decomposition into systems, subsystems, and components. As shown in Figure 2.5, this decomposition can be represented as an extended V-model. The hierarchical decomposition scheme with level indices (n), (n+1), and (n+2) is based on the author's previous publication [2, pp.12-13]. At the highest level (Level n), the aircraft is treated as a single integrated system. This system can be subdivided into multiple subsystems at Level (n+1), such as propulsion, avionics, or the environmental control system. If a subsystem remains too large or complicated, it can again be decomposed into sub-subsystems at Level (n+2), and so forth. For instance, the propulsion may be broken down at Level (n+2) into the engine core, nacelle, and control units. The hierarchical decomposition can be continued recursively until the required level of detail is reached. This hierarchical refinement is illustrated in the Figure 2.5 using indices i and j. The index i denotes the number of subsystems at Level (n+1), while j represents the number of sub-subsystems at Level (n+2). Thus, subsystem (n+1).1 can itself be divided into sub-subsystems (n+2).1.1, (n+2).1.2, ..., up to (n+2).1.j [2, pp.12-13]. At each level, the same development steps recur: requirements analysis, functional decomposition, logical architecture, and physical architecture. Decisions made at higher levels cascade downward, while refinements at lower levels feed back upward.

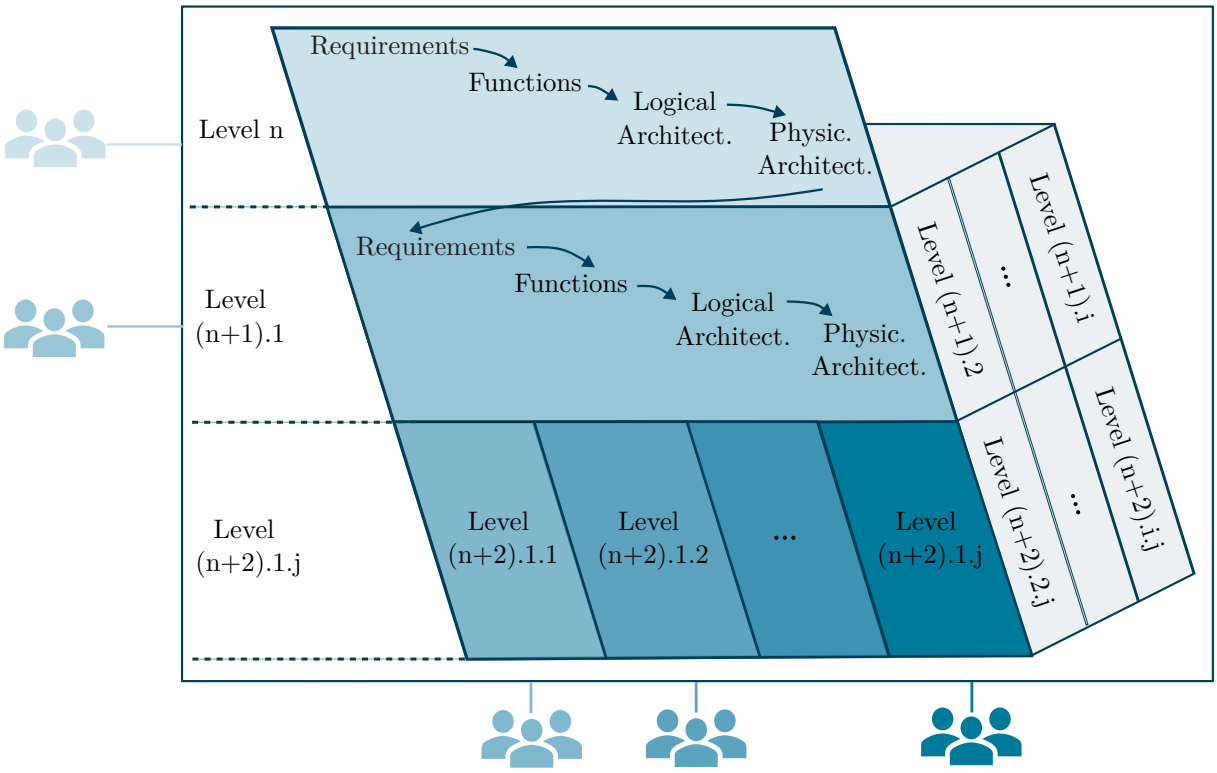

Figure 2.5.: Representation of the extended V-model for multi-level development [2]

Second, the multidisciplinary development intensifies this challenge. Specialized teams work in parallel on structure, aerodynamics, systems, and avionics. While this specialization improves efficiency, it often obscures cross-domain interactions. The Airbus 380, for example, required coordination across more than 70 functional subsystems [13, p.1]. This way of development creates knowledge silos with domain-specific terminologies. A structural engineer knows, for example, her "system" perfectly but has only a limited understanding of engine integration, even though these systems have several interactions. While specialization improves efficiency in individual domains, system incompatibilities often remain undetected until late in the integration testing process. This can lead to costly rework cycles [80, p.34].

Third, documentation intensity amplifies both hierarchical and disciplinary challenges. Despite advances in MBSE, aircraft programs generate vast collections of textual specifications, compliance documents, and certification evidence [89, p.4]. Digital repositories improve accessibility but do not address the fundamental difficulty: extracting, integrating, and applying knowledge across system levels and disciplines. Moreover, documentation from completed programs becomes the baseline for subsequent ones. Next-generation developments or evolutionary variants begin with a systematic assessment of what can be adopted, adapted, or newly designed. This requires engineers to navigate legacy documentation, certification records, and lessons-learned databases.

This analysis phase is inherently iterative and repetitive. Documents are reviewed in multiple cycles, cross-compared across disciplines, and continuously updated as assumptions evolve. Engineers must trace interdependencies, safety implications, and certification requirements before proposing design solutions. Because around 85% of lifecycle costs are determined during early design phases [84, p.2], inefficiencies at this stage directly translate into prolonged schedules and escalating costs. The problem is compounded by the

shortage of qualified engineers, with projections indicating up to 390,000 vacancies in Germany alone by 2029 [30], as mentioned earlier in Chapter 1.

Taken together, aircraft development exemplifies a domain in which manual knowledge processing is a central challenge. As shown in the preceding analyses of document-based and model-based approaches (Sections 2.1 and 2.2), representation and traceability have improved considerably, yet the effort of extracting, consolidating, and reusing information across technical documents remains high. These limitations become particularly evident in the context of aircraft development (Section 2.3), where engineers must navigate complex, multidisciplinary documentation. A repetitive, document-, and resource-intensive task. These characteristics make aircraft development an ideal testbed for evaluating whether GenAI-augmented approaches can accelerate knowledge synthesis.

2.4. Artificial Intelligence

Artificial Intelligence (AI) has evolved into a diverse field encompassing a wide range of techniques for data-driven reasoning and decision-making. Within the context of aircraft development, AI methods offer new opportunities to accelerate the analysis of complex document repositories while maintaining traceability and compliance. The following sections introduce the technological foundations. From AI over Machine Learning, Deep Learning, and GenAI to LLMs, and position them within the hierarchical structure of the AI landscape that underpins this dissertation.

2.4.1. AI Technologies

Understanding the fundamentals and development of AI technologies provides an important context for evaluating their application in knowledge-intensive areas of engineering. A historical overview of the development of AI can be found in Appendix A.3 on page 184. At its core, the field of AI can be conceived as a hierarchical framework in which each level builds on and extends the capabilities of the previous one, see Figure 2.6.

AI represents the broadest framework, defined by ISO 22989:2022 as "*a technical and scientific field devoted to the engineered systems that generates outputs such as content, forecasts, recommendations, or decisions for a given set of human-defined objectives*" [91, p.vii].

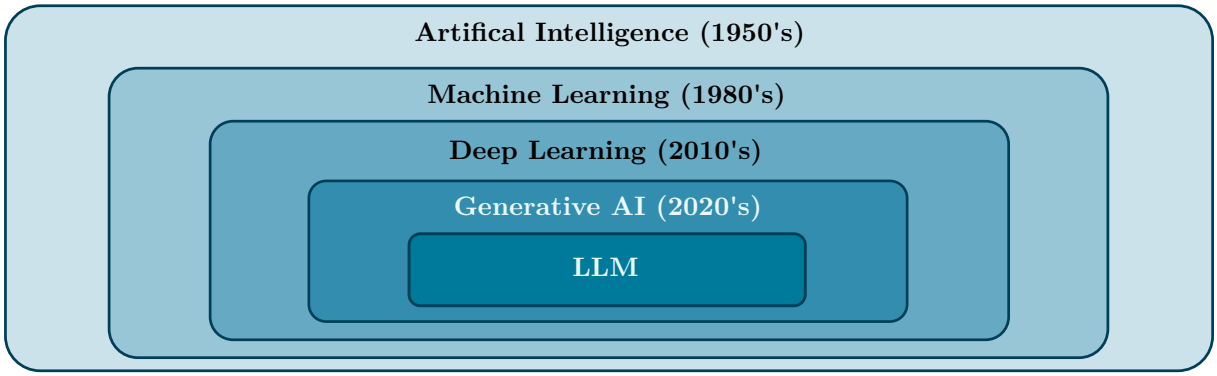

Figure 2.6.: Hierarchical categorization of GenAI and LLMs within the AI spectrum, adapted from [90]

Within this domain, Machine Learning (ML) involves models that derive predictions or decisions based on data without being explicitly programmed for each task. Classical methods include linear and logistic regression, decision trees, or random forests [90]. The use of artificial neural networks, whose structure is loosely inspired by the human brain, has been a key driver of recent advances. These networks are particularly effective for pattern recognition and complex data analysis. ML has been successfully applied across domains such as text classification, Natural Language Processing (NLP), computer vision, medicine, fraud detection, and autonomous systems [92, p.2], with performance strongly dependent on data quality and algorithmic design [93, p.2].

Deep Learning (DL) is a subdomain of ML that employs multi-layer neural networks to learn hierarchical data representations. Through gradient-based optimization, DL models adjust their internal parameters to minimize a loss function and thereby improve predictive accuracy. DL methods have transformed image, speech, and NLP by automatically learning relevant features from input data [94, pp.436–439]. For example, a DL could analyze a photo of a market, determine that there are multiple people present, and produce a caption like this: "*A group of people shopping at an outdoor market.*" In doing so, the model maps visual patterns, such as form, colors, and spatial layout, to linguistic concepts and full sentences [94, p.440].

Generative AI (GenAI) builds on these DL foundations. Instead of predicting labels or categories, GenAI models learn probability distributions over data to generate new content. These models are trained on extensive and heterogeneous datasets and can be adapted across tasks without complete retraining [95]. According to Munro et al. [96, p.460], GenAI models generate new data across different modalities, including text, images, and video, by capturing contextual relationships and implicit rules within training data. For example, image generation requires an understanding that paintings in interior scenes typically appear on walls rather than floors. These relationships are approximated as multi-dimensional probability distributions, which the model learns from extensive document databases, also referred to as corpora in this dissertation [96, p.460]. This principle applies equally to programming code, scientific texts, or medical questionnaires. While the scaling of datasets leads to increasingly sophisticated outputs, the approximations remain imperfect and reflect the challenges of modeling complex relationships in the real world [96, p.460].

Large Language Models (LLMs) are the most prominent implementation of GenAI[5]. They demonstrate sophisticated abilities in understanding, interpreting, and generating natural text with human-like coherence across diverse domains [33, p.15]. The market for LLMs has developed rapidly in recent years, characterized by an increasing number of models. Figure 2.7 illustrates the release timeline of models from Anthropic, OpenAI, and Google, which were selected to represent the most influential LLM providers examined in this dissertation. Other providers were omitted to maintain readability. The figure thereby illustrates recent development trends, but does not reflect the full breadth of the market. The timeline reveals an apparent acceleration in development cycles, particularly since 2022. Beyond these established providers, a broad net of additional LLM developers exists, including both commercial companies and open-source projects. According to Thompson [97], more than 100 organizations have released in total over 660 different LLMs. This breadth of actors contributes to a competitive environment characterized by shorter release cycles and steadily improving model capabilities. The high level of dynamism underscores the central role of LLMs for the further development of AI in technical application domains.

As can be seen in Figure 2.7, a distinct category of so-called *reasoning LLMs* has emerged [99, p.3], including models such as OpenAI's o1 and Claude-3.7-sonnet. These models

[5]Beyond text, GenAI also includes models for image synthesis, as well as audio, video, and multimodal generation systems that integrate multiple data modalities [98, p.1].

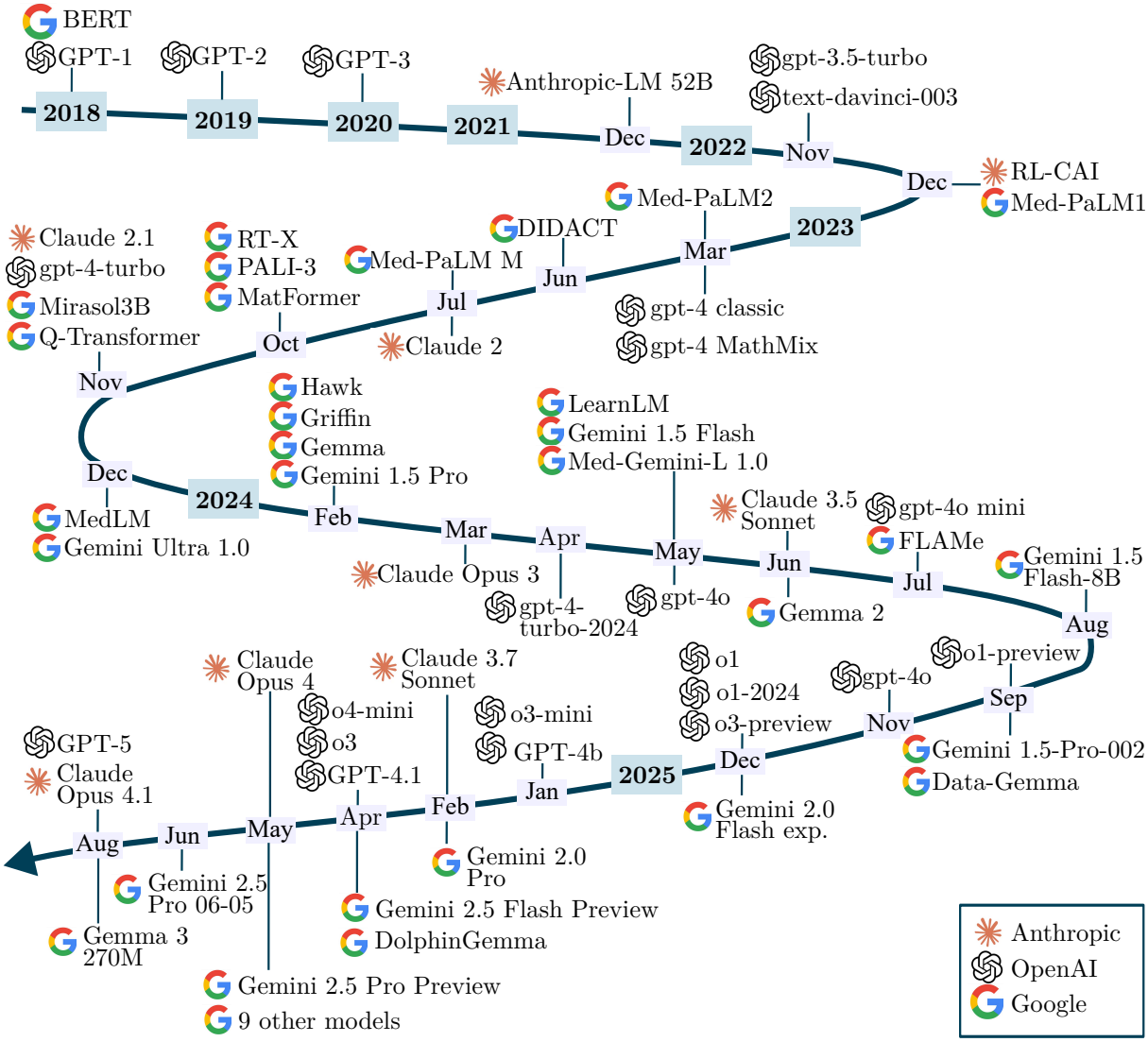

Figure 2.7.: Development timeline of selected LLMs from three major providers: Anthropic, OpenAI, and Google, from 2018 to 2025 (own representation based on the data from Thompson [97])

build upon traditional LLMs by explicitly modeling intermediate reasoning steps. This enables a more structured approach to problem-solving in tasks that require logical inference, mathematical deduction, or strategic planning [99, p.3], [100, p.1]. While such approaches achieve notable gains in reasoning benchmarks and domain transfer, they also introduce practical challenges. They demand higher computational effort, respond more slowly, and may produce unnecessarily long or inconsistent reasoning chains [101, p.8]. Accordingly, their advantages are most evident in applications where traceability and correctness are prioritized over inference speed.

2.4.2. Large Language Models

An LLM can be understood as a statistical language model trained on vast collections of text [102, pp.187-190]. The goal is to capture linguistic structures and semantic relations

to generate coherent text [103, p.7]. The development of such models follows a systematic sequence of steps, as illustrated in Figure 2.8.

To build an LLM, the first step is the segmentation of text into smaller linguistic units, known as *tokens*. Tokens can be whole words, characters, or parts of words. LLMs predominantly rely on subword tokenization algorithms such as Byte-Pair Encoding (BPE) [104], which construct their vocabulary in a data-driven way. Frequent character sequences are merged into single tokens. As a result, common words are often represented as a single token, whereas rare words are split into multiple subword units. Since these units are derived from statistical frequency patterns rather than linguistic principles, they may, but do not necessarily, correspond to morphemes[6]) [102, p.13]. For example, the sentence "*The capital of France is Paris*" may be segmented into the six tokens "*The*", "*capital*", "*of*", "*France*", "*is*", "*Paris*", as can be seen in Figure 2.8 [105, p.20]. This token-based representation replaces the original text with a sequence of discrete units that serves as the basis for all subsequent processing steps.

In the next step, every token is transformed into a numerical vector, known as token *embedding*, which reflects its meaning and relation to other tokens in a multi-dimensional space. In this space, words with similar meanings lie close together, while unrelated words are farther apart [102, p.180], [106]. For instance, the vectors for "*Paris*" and "*France*" would occupy nearby regions, whereas "*car*" would be positioned much farther away, as can be seen in Figure 2.8. In this way, language can be represented mathematically. Distances and directions in the embedding space correspond to linguistic relationships such as similarity, analogy, or context. This step provides the numerical foundation that enables LLMs to process and reason about language.

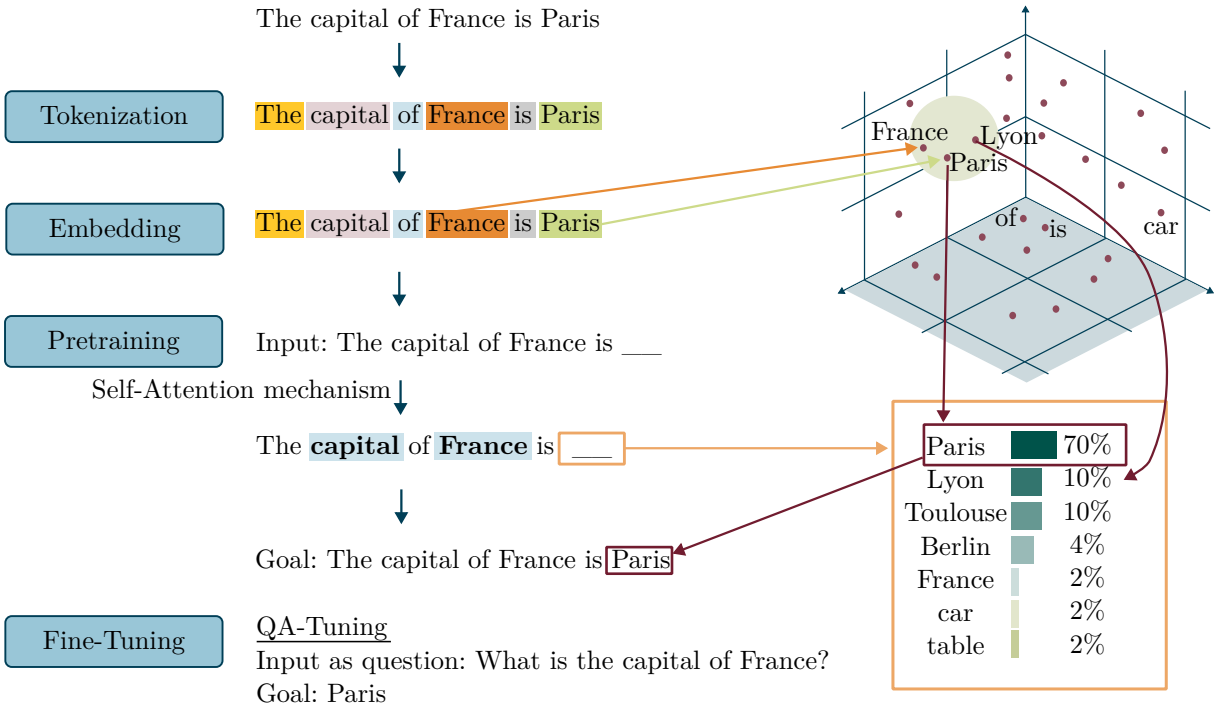

Figure 2.8.: Schematic overview of the functionality of an LLM

[6]Jurafsky and Martin [102, p.8] define a morpheme as "*[..] the minimal meaning-bearing units into which words can be divided*". Morphemes may occur independently as free morphemes (e.g., "*house*") or bound morphemes (e.g., prefixes or suffixes). For example, the word "*dogs*" consists of the stem "*dog*" and the plural morpheme "*-s*".

During pretraining, the model learns statistical patterns in language by predicting the next token in a sequence [102, p.171]. This process, known as *autoregressive training*, forces the model to capture grammatical structures, word dependencies, and contextual relationships. Formally, the model estimates a probability distribution over possible continuations of a sentence [33, p.11]. To illustrate, consider the input "*The capital of France is* ___". The model assigns high probability to "*Paris*," lower probability to "*Lyon*," and almost zero probability to unrelated words such as "*car*." By repeating this prediction task across trillions of examples, the model gradually develops an internal statistical picture that characterizes the relationships between words in natural language. However, as Brown et al. [33, p.11] point out, such models can also inherit undesired patterns from their training data, such as biased or stereotypical associations.

At their core, LLMs employ the *transformer architecture* developed by Vaswani et al. [107], which marked a turning point in NLP. Unlike earlier neural networks that processed text step by step, transformers can analyze all tokens of a sentence simultaneously. Their key mechanism, called *self-attention*, allows the model to evaluate which words are most relevant to one another in the context of meaning [107, p.6]. For example, when processing the prompt "*The capital of France is* ___," the model learns to focus on the tokens "*capital*" and "*France*" to predict "*Paris*" with high confidence. This parallel processing makes training vastly more efficient, enabling the model to represent contextual relationships over long text spans and overcoming the limitations of earlier architectures, such as vanishing gradients. Because the transformer mechanism is domain-agnostic, the same principle can also be applied to other data types such as images, video, audio, or sensor data, thereby unifying previously separate AI modalities. These properties, like parallelization [107, p.2], scalability, and flexibility, form the technical foundation of modern large-scale language models such as GPT, Claude, and Gemini [108, p.4].

After pretraining on general text corpora, an LLM is usually adapted to specific tasks or domains through an additional training phase known as *fine-tuning*. In this step, the pretrained model continues learning on smaller, task-oriented datasets, such as technical documentation, dialogue examples, or domain-specific question-answer pairs, so that it can perform more specialized functions. A second and increasingly important refinement step is *Reinforcement Learning from Human Feedback* (RLHF) [109, p.2]. Here, humans compare alternative model outputs and rank them according to criteria such as helpfulness, accuracy, and safety. These rankings are used to train a reward model that captures human preferences. Subsequently, the LLM is optimized through reinforcement learning using this reward signal, aligning its behavior more closely with human expectations. This approach makes the system's responses more consistent with user intent and reduces undesired effects such as factual errors or biased formulations. A well-known example is *InstructGPT*, which, despite having far fewer parameters than GPT-3, was preferred by users for its accuracy, clarity, and reduced toxicity in responses [109, p.1]. Together, fine-tuning and RLHF transform general-purpose language models into tools that can interact safely and effectively within specialized technical or industrial contexts.

Prompt Engineering

Prompt engineering refers to the specific and iterative design of inputs to LLMs to produce precise and useful results [110, p.22]. A *prompt* is defined as "*an input to a Generative AI model that is used to guide its output*" [111, p.5].

Methodologically, prompt engineering follows a recurring cycle: (1) submit a prompt to the LLM that generates an answer, (2) evaluate the result, and (3) revise the prompt template until the desired criteria are met [111, p.7]. Typical prompt components include an explicit instruction (directive), optional examples, guidance on the desired output

format, a role description, and additional context information [111, pp.5–6]. The main components and representative prompt types are summarized in Box 1.

Core prompting techniques include zero-, one-, and few-shot prompts. In a zero-shot prompt, the LLM receives only a task description like "*What is the capital of France?*" and has to generate the answer solely on its internal knowledge. In one-shot and few-shot prompts, by contrast, one or several examples are added to illustrate the expected procedure or output structure [110, pp.64-66]. For more complex tasks, the Chain-of-Thought (CoT) technique has proven to be a practical approach. Here, the prompt instructs the LLM to proceed step by step and to make its intermediate reasoning explicit [110, p.67], [111, p.12]. Recent literature distinguishes more than fifty text-based prompting techniques [111, pp.1, 9]. For practical applications, Koch et al. [110, p.74] recommend beginning with simple prompts, evaluating outputs systematically, and considering model limitations such as context window size or hidden system prompts, particularly when constructing longer prompt chains. Box 1 provides concrete examples of these techniques, partially adapted from Koch et al. [110] and Schulhoff et al. [111].

Beyond such task-focused prompting, LLMs can also be provided with broader background information directly within the same prompt. This may include supplementary text, technical papers, or system models, enabling the LLM to adapt its response to a specific context. For instance, a paper can be supplied with the instruction to summarize it rather than to answer a factual question. In this dissertation, this principle is referred to as *in-context prompting*, a general mechanism by which LLMs condition their output on the information provided within the prompt rather than relying solely on their pre-trained knowledge.

Box 1: Example of Different Prompt Techniques

Components of a Prompt

Directive	`Give me three recipes for dinner`
Output Formatting	`Summarize the data into a csv file`
Style Instruction	`Write a easily understandable text about Paris`
Role	`Pretend to be an engineer with more than ten years of experience`

Prompt Techniques

Zero-Shot:	`What is the capital of France?`

One-Shot:	`Determine the language style of the request. Example:`
	`Request: Can you help me? Style: Friendly`
	`Request: Why don't you answer me?`

Few-Shot:	`Determine the language style of the request.`
	`Here are a few examples:`
	`Request 1: Can you help me? Language style: Friendly`
	`Request 2: Help me! Language style: Demanding`
	`Request 3: Answer me now. Language style: Unfriendly`
	`Request: Why don't you answer me?`

Chain-of-Thought:	`Proceed step by step.`

Evaluation of LLM Outputs

To evaluate the performance of LLMs, standardized benchmarks are employed. According to Kistowski et al. [112, p.333], a benchmark is a comparative standard used to evaluate the performance of systems or processes based on defined reference tasks. In the context of LLMs, benchmarks provide structured test collections that define *what* the model should

accomplish (e.g., classification, summarization, or reasoning), whereas evaluation metrics determine *how well* the model performs these tasks [113].

Benchmarks consist of structured datasets and well-defined challenge tasks, each accompanied by corresponding reference solutions. Different evaluation modes are commonly applied. In the *zero-shot* setting, the model must solve a task without seeing examples, thereby testing its ability to generalize to new scenarios. The *few-shot* setting provides a handful of demonstrations, assessing how well the model can infer patterns from limited information. Finally, in the *fine-tuned* setting, the model is explicitly trained on task-specific data to maximize performance on the corresponding benchmark [113].

Benchmarks thus define the *task framework* for evaluation. As an illustrative example, Table 2.1 presents publicly available benchmark results that compare GPT-4.1 and the reasoning-oriented model GPT-o1 [114]. The results highlight that different benchmarks assess distinct capabilities, such as general reasoning, scientific reasoning, code generation, or multimodal understanding, rather than providing a comprehensive measure of overall intelligence.

Table 2.1.: Benchmark results comparing GPT-o1 and GPT-4.1 with data from [114]

Benchmark	**GPT-o1**	**GPT-4.1**	**Benchmark description**
MMLU	92.3%	90.2%	Massive Multitask Language Understanding: benchmark evaluating broad reasoning and factual knowledge across 57 academic subjects covering STEM, humanities, law, and medicine. Widely used for assessing the general reasoning ability of LLMs [115].
GPQA	77.3%	66.3%	Graduate-level Google-Proof Q&A Benchmark: dataset of 448 expert-written multiple-choice questions in physics, chemistry, and biology designed to test advanced scientific reasoning beyond typical graduate-level difficulty [116].
HumanEval	92.4%	n/a	Measures code generation and functional correctness on Python programming problems. It assesses a model's ability to produce working software from natural-language prompts [117, p.4].
MMMU	78.2%	74.8%	Massive Multi-discipline Multimodal Understanding: large-scale benchmark combining text, image, and diagram reasoning across 30 subjects from six academic disciplines, testing multimodal perception and higher-order reasoning [118].

Once an LLM has produced an answer for a specific benchmark task, its output is evaluated using quantitative metrics that compare the predicted answer to the benchmark's reference solution. The following overview summarizes the most commonly applied evaluation criteria, based on the descriptions provided in [113]:

- Accuracy, precision, recall, and F1-score: These metrics are mainly used for classification or question-answering tasks. Accuracy measures the overall share of correct answers. Precision indicates the proportion of predicted items that are correct, while recall measures how many of the true reference items the model identified.

The F1-score combines both perspectives of precision and recall into a single value and is calculated as

$$F_1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{2.1}$$

Example: For the benchmark question "*Which fuel is used in a fuel cell system?*" with the reference answer "*Hydrogen*", a model answering exactly "*Hydrogen*" would achieve: Precision = 100%, Recall = 100%, and F1 = 100%. If the model answers "*Hydrogen and Oxygen*", Precision drops to 50% because Oxygen is not in the reference, while Recall remains 100%. The resulting F1-score is about 67%.

- Exact Match: This metric is stricter and requires that the model's answer exactly match the reference text. Any deviation, even if semantically equivalent, is considered wrong.
  Example: If the reference answer is "*Hydrogen*", the model must produce exactly that word. An answer like "H_2" would be counted as incorrect, even though it means the same.

- Perplexity: Perplexity measures how effectively a language model predicts the next word in a sequence. A lower perplexity indicates greater confidence in its predictions and reflects that the text is more predictable or linguistically consistent for the model.
  Example: For the sentence beginning "*The aircraft flies through the ...*", a well-trained model will strongly favor completions such as "*air*" or "*sky*", resulting in low perplexity. If it assigns similar probabilities to many unrelated words, the perplexity will be high.

- BLEU, ROUGE, and METEOR: These metrics are commonly used to evaluate tasks such as translation and summarization. They measure the overlap between short word sequences (so-called *n-grams*[7]).
  Example: If the reference summary reads "*The aircraft engine uses hydrogen cooling*", and the model produces "*Hydrogen cooling is used in the aircraft engine*", the n-gram overlap will be high because the same words appear in both sentences, even though the phrasing differs.

- Human evaluation: Because automatic metrics cannot capture everything, reviewers are also used to judge model outputs. They often assess aspects such as coherence, clarity, and factual correctness. This approach provides richer insight but is time-consuming and subjective.
  Example: A human judge might consider a summary to be clear and technically correct, even if automatic metrics assign a low score because the wording differs from the reference.

In practice, LLM evaluation follows a structured workflow:

1. A benchmark provides a set of tasks and corresponding reference answers.
2. The LLM generates responses for all tasks (zero-shot, few-shot, or fine-tuned).
3. Evaluation metrics are applied to compare LLM outputs with reference answers and compute scores.

[7] An *n-gram* is a sequence of n consecutive words. A 1-gram (unigram) is a single word, such as "*aircraft*", a 2-gram (bigram) is a word pair such as "*electric system*", and a 3-gram (trigram) is a three-word sequence such as "*more electric aircraft*". The metrics count how many of these n-grams from the model output also appear in the reference text. A higher overlap results in a higher score, reflecting greater surface-level similarity. However, these metrics do not evaluate whether the generated content is semantically accurate or factually correct.

4. These scores quantify the model's performance in the benchmark domain.

Because no single benchmark or metric captures every dimension of performance, multiple benchmarks and metrics are typically combined for a reliable overall assessment. All in all, benchmarks are helpful because they provide quantitative indicators that make a model's progress measurable and highlight where it excels and where improvement is needed. However, numbers alone capture only part of it. Increasingly, qualitative dimensions (coherence, relevance, and semantic adequacy) are assessed via human rating to complement automatic scores. A balanced combination of quantitative and qualitative metrics is therefore considered essential for a reliable overall assessment.

While these metrics enable comparability across tasks, they also have fundamental limitations, as discussed by Caballar and Stryker [113]. First, most benchmarks rely on generic datasets and transfer poorly to specialized application domains. Second, they provide only a static snapshot of current model capabilities and do not capture future development potential. Third, they risk overfitting when models are directly or indirectly trained on benchmark data, which can yield artificially high scores without genuine generalization. Recent studies confirm these shortcomings: Zhang et al. [119] demonstrate that classical n-gram metrics show a low correlation with human evaluations, whereas semantic and fact-based measurements achieve substantially higher alignment. Similarly, Hu and Zhou [120] emphasize that future evaluation frameworks must be multidimensional, combining automatic, semantic, and human-in-the-loop assessments with a stronger focus on factuality, reasoning, and coherence. As a result, conventional benchmarks provide a coarse orientation but are increasingly insufficient to assess the practical utility of LLMs in domain-specific applications such as aircraft engineering.

Potential and Early-Stage Applications of LLMs in Engineering Documentation

LLMs have demonstrated remarkable capabilities for processing and generating technical text in the engineering field. Their potential is particularly evident in document-centered engineering tasks, where large volumes of textual information must be analyzed, summarized, and reformulated. The following section outlines key capabilities of LLMs and illustrates how recent studies have explored or proposed their application to engineering contexts relevant for aerospace documentation and knowledge processing.

Knowledge extraction and summarization: Mustapha [121, p.15] provides a comprehensive survey of emerging LLM applications across mechanical engineering and manufacturing. The cited and reviewed studies commonly evaluate LLMs by supplying textual engineering documents (e.g., design specifications or simulation reports) through the model's context window and then testing their ability to extract, summarize, or retrieve relevant information. Examples include the automatic extraction of functional requirements from design documents, knowledge retrieval against design requirement documents, and semantic interpretation of simulation data. These examples demonstrate how LLMs can be utilized to extract and reformulate knowledge from technical documentation. Mustapha concludes that such approaches show promising efficiency gains, but remain limited by dataset availability and a lack of standardized evaluation protocols [121, pp.28-29]. Overall, these text analysis capabilities offer a solution to one of the most significant challenges in aircraft development: the need to summarize information from extensive document sets to maintain a consistent and up-to-date knowledge base.

Requirements understanding and extracting: In the context of Requirements Engineering (RE), Zadenoori et al. [122] published a systematic review that provides a comprehensive overview of 74 studies from 2023 and 2024 examining how LLMs are applied across RE tasks. Their analysis shows that research focuses predominantly on text-heavy and cognitively demanding activities, such as requirements elicitation and validation. At the same

time, traditional NLP-based approaches that focus on defect detection, such as identifying ambiguities, inconsistencies, or incomplete statements, and classifying requirements into predefined categories, have become secondary. The reviewed studies cover all major RE phases, from elicitation and analysis to modeling and validation, and primarily use system and software requirement specifications, user feedback, and regulatory documents as input to the LLM (context window). GPT-based models dominate, often in zero- or few-shot prompting setups, and are typically evaluated in controlled laboratory environments. Zadenoori et al. conclude that the field is still in an exploratory stage, with fragmented methods and limited industrial validation, yet highlight that LLMs already show strong potential for knowledge extraction, documentation assistance, and traceability within RE workflows.

Design reasoning and conceptual support: Chiarello et al. [123] conducted a large-scale bibliometric and qualitative analysis of 15,355 publications to identify where LLMs could contribute within the engineering design process. Their review categorized potential contributions into three functional groups (*generate*, *evaluate*, and *describe*) and mapped them to different design phases, concluding that approximately 40% of analyzed studies contain tasks that could, in principle, be supported by LLMs. The paper highlights promising directions for using LLMs in conceptual exploration, design evaluation, and documentation of design rationales, while emphasizing the continued need for human oversight and validation [123, pp. 1966–1967]. Although these findings remain prospective rather than demonstrative, they suggest that LLMs may, in the future, support conceptual reasoning and architectural exploration in engineering domains such as aerospace.

Documentation assistance and efficiency gains across domains: Cross-domain studies further illustrate the broader potential of LLM-based text processing in intensive documentation environments. In healthcare, LLMs reduced the documentation time within pilot projects by 30-40% (e.g., transcription, summarization), while in the automotive industry, compliance-reporting costs decreased by 10-15% [124, pp.11-12]. Similar efficiency gains are anticipated in other engineering domains, particularly for technical documentation, requirement management, and maintenance logs. In software development, LLMs are utilized to document design decisions, derive architectures from textual requirements, generate component descriptions, and facilitate model-to-code mappings [125, p.6]. Moreover, Zhang et al. [126, p.6] demonstrate that LLMs can accelerate literature reviews by condensing and synthesizing scientific sources. A capability directly applicable to the automated analysis of engineering standards, patents, and system documentation.

Taken together, the reviewed studies indicate that LLMs hold substantial potential for extracting, structuring, and reformulating knowledge across technical domains. Their text-analytic and reasoning capabilities make them particularly relevant for documentation- and regulation-intensive disciplines such as aircraft engineering.

Challenges for Engineering Applications

Despite impressive advances, recent literature also highlights several limitations that constrain the direct applicability of LLMs in technical engineering domains. As outlined in Chapter 1, general limitations of LLMs, such as their restricted context window, the lack of access to domain-specific data sources, and the tendency to generate hallucinated information, form the baseline challenges for their reliable use. Beyond these generic constraints, recent studies highlight additional, engineering-specific issues that further limit their applicability in the aircraft contexts.

The first challenge is that LLMs are trained with datasets that have specific temporal boundaries, so-called “knowledge cutoffs.” Those cutoffs represent the latest date of available training information. Nonetheless, research by Cheng et al. [127] show that these

cutoffs are more problematic than commonly assumed. The discrepancy between stated and actual knowledge cutoffs arises from deduplication problems and temporal distortions in the training data, with available current information often falling significantly short of the stated cutoffs. In aircraft applications, this means that LLMs may provide outdated information on rapidly evolving technologies, such as sustainable aircraft fuels, electric propulsion systems, or updated certification standards. In the system design process, this implies that models cannot rely solely on training corpora, but must continuously incorporate new and authoritative documents.

A second challenge concerns the traceability. According to Zhang et al. [128], LLMs generate information without indicating its origin when operating without an explicit context. In such cases, the model cannot reference specific sources, since its responses are derived from statistical patterns learned during pretraining rather than from identifiable documents. This lack of source transparency makes it difficult to verify statements against authoritative standards or validated data.

Third, studies also reveal significant weaknesses in LLMs' ability to perform mathematical calculations and interpret technical notation, both important during the development of new technical systems. In a study from 2024, Mirzadeh et al. [129] documented up to 65% performance degradation in mathematical tasks when additional text was inserted into the problem statements. This is directly relevant to processing technical documents, where critical calculations are often embedded within extensive contextual information. Specifically, LLMs may struggle to accurately interpret and perform complex calculations in technical standards, convert units between systems, or precisely capture mathematical relationships in system specifications. These limitations are particularly critical because the inability to perform necessary calculations creates fundamental gaps in system design: without reliable mathematical results, no well-founded design decisions can be made.

In summary, the analysis reveals a fundamental tension. On the one hand, LLMs excel at summarizing heterogeneous information, structuring it into coherent narratives, and supporting cross-document comparisons. Those capabilities are highly relevant for the early analytical phases of aircraft design. On the other hand, they are constrained by a limited context window size, a lack of access to domain-specific and proprietary data sources, are prone to outdated knowledge, and have a tendency to produce unverifiable information. As a result, a standalone use of LLMs in the aircraft field is not feasible.

The central question is therefore: *How can the synthesis capabilities of LLMs be applied while ensuring that outputs remain current, verifiable, and technically precise?*

This challenge does not suggest abandoning LLMs, but points to the need for systematic augmentation. A promising solution lies in anchoring language model outputs to authoritative technical sources, with explicit citations and transparent reasoning. This requirement leads directly to Retrieval-Augmented Generation (RAG), which couples generative models with curated document repositories.

2.4.3. Retrieval-Augmented Generation

The limitations of standalone LLMs in technical domains highlight a central requirement: generated content must rely on reliable and verifiable sources. RAG meets this requirement by combining the generative capabilities of LLMs with targeted retrieval from curated corpora [130, p.552], [131, p.1]. Initially introduced by Lewis et al. [132] in 2020, RAG combines two capabilities: (1) semantic retrieval of context passages from external knowledge bases and (2) generation of responses conditioned on these retrieved passages.

For regulated engineering domains such as aircraft development, this architecture is attractive because it preserves the fluency and synthesis of LLMs while adding verifiability and traceability. Conceptually, a RAG system[8] operates in two stages, see Figure 2.9. A *data preprocessing* phase prepares the domain corpus (collection, cleaning, chunking, and embedding), followed by a *retrieval*→ *augmentation*→ *generation* loop that grounds answers in the retrieved evidence during query execution. The intermediate vector database serves as the connecting layer between both stages, storing the embedded representations and enabling efficient semantic retrieval. The following sections detail both stages and summarize observed benefits and current limitations in engineering use.

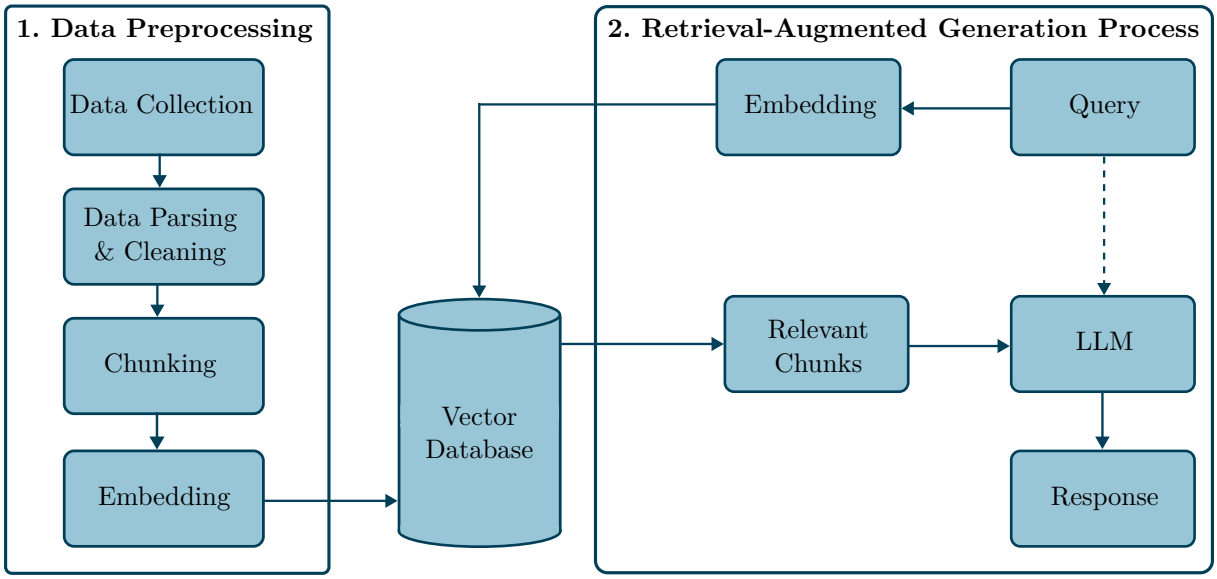

Figure 2.9.: Architectural overview of a typical RAG system, adapted from [133]

Phase 1: Data Preprocessing

The data preprocessing phase defines the domain-specific corpus on which all subsequent processes depend [134, p.6]. It begins with the systematic collection of data relevant to the application context. Depending on the use case, this may include technical documentation, regulatory standards, academic publications, and, within companies, proprietary sources such as internal reports, process descriptions, and operational guidelines.

Once collected, the data is processed in three steps: parsing, cleaning, and chunking [134, p.6]. During parsing, the source files (e.g., PDF, DOCX, HTML) are converted into a machine-readable representation. Structural elements such as lists, tables, and paragraphs are preserved to support later segmentation and retrieval quality [134]. In the cleaning step, non-content elements that would disturb segmentation, such as headers, footers, page numbers, watermarks, reference lists, or embedded metadata, are removed. In the next step, the cleaned text is segmented into semantically coherent units, so-called chunks. The chosen chunk size is critical: overly large chunks reduce retrieval precision, whereas overly small chunks fragment context. Effective chunking, therefore, balances granularity and coherence to prepare the corpus for robust embedding [135].

[8] Although the underlying structure is conceptually closer to a pipeline or architecture, the term *RAG system* has become widely established in the literature to denote implementations of *Retrieval-Augmented Generation*. This wording emphasizes the integrated interaction between retrieval and generation components rather than their sequential processing.

The final step involves embedding and indexing, where each chunk is transformed into a numerical vector representation using specialized embedding models. These vectors are then indexed in a vector database optimized for high-dimensional similarity search. A schematic illustration can be seen in Figure 2.10. Information with similar meanings is located closer together in this space, regardless of the specific vocabulary used. For instance, chunks discussing "*Paris*" and "*the Eiffel Tower*" would occupy nearby positions in this vector space, as both refer to semantically related concepts about France.

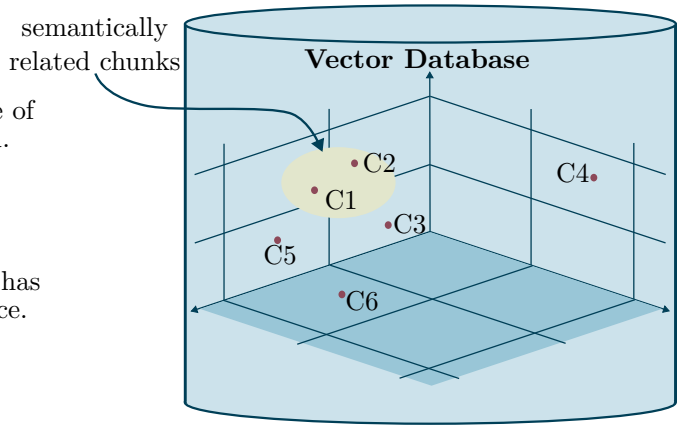

Figure 2.10.: Illustration of a semantic vector space in a RAG system. Each point represents a text chunk, and spatial proximity reflects semantic similarity.

This vectorization process enables semantic rather than purely lexical matching, allowing the system to retrieve relevant information that is conceptually related even if the terminology differs between the query and the source material [135]. In this way, the approach transcends traditional keyword matching by capturing deeper conceptual relationships. For instance, a chunk describing "*Lyon as a major town in eastern France*" may still be considered relevant to a query about "*French cities*", even without direct lexical overlap, since their vector representations encode similar semantic concepts.

Phase 2: Retrieval-Augmented Generation Process

The characteristic of RAG systems is their design for single-query interactions, where each query is processed individually to generate a focused response. This design pattern has made RAG systems exceptionally successful for chatbot applications and technical support systems across different industries, where users seek specific answers to individual questions.

When responding to a question from a user, the RAG system executes a systematic sequence. The following explanation of the RAG workflow is based on the conceptual description by Klesel and Wittmann [130, p.553]:

1. Query Definition: The user submits a request/question, also known as a "*query*" (see Figure 2.11).
2. Query Processing: The user's query undergoes the same embedding procedure applied to the knowledge base, preserving semantic intent while enabling comparison with stored information.
3. Retrieval: The system performs similarity calculations, typically employing cosine similarity metrics, to compare the query with stored document passages in the vector database. Based on this ranking, it retrieves the top-k most similar chunks,

where k determines the trade-off between concise (small k) and more complete but potentially noisier (large k) context. During this stage, the similarity between the embedded query and the stored vectors is computed using mathematical distance metrics that quantify semantic proximity. The most common approaches, as shown schematically in Figure 2.11, include:

- Cosine Similarity: Measures the angle between two vectors, emphasizing directional alignment rather than absolute magnitude. It is the standard choice in most text-embedding applications.
- Euclidean Distance: Calculates the absolute distance between two vectors in the high-dimensional space, considering both direction and length.
- Manhattan Distance: Sums the absolute differences of all vector dimensions (L1 norm), making it more robust to outliers.
- Chebyshev Distance: Considers only the largest single-dimensional difference.

Depending on the metric, the resulting similarity landscape changes slightly.

4. Context Assembly: The retrieved chunks are integrated into a unified context that provides a factual foundation for response generation, often emphasizing authoritative sources and structured information.
5. Response Generation: The LLM generates the output based on both the original query and retrieved information, combining fluent articulation with factual precision from retrieved content.
6. Response: The system delivers the response with appropriate attribution, typically including references to source documents that enable verification. This traceability represents a critical feature for engineering applications, where decisions based on generated information may have significant safety, regulatory, or economic implications.

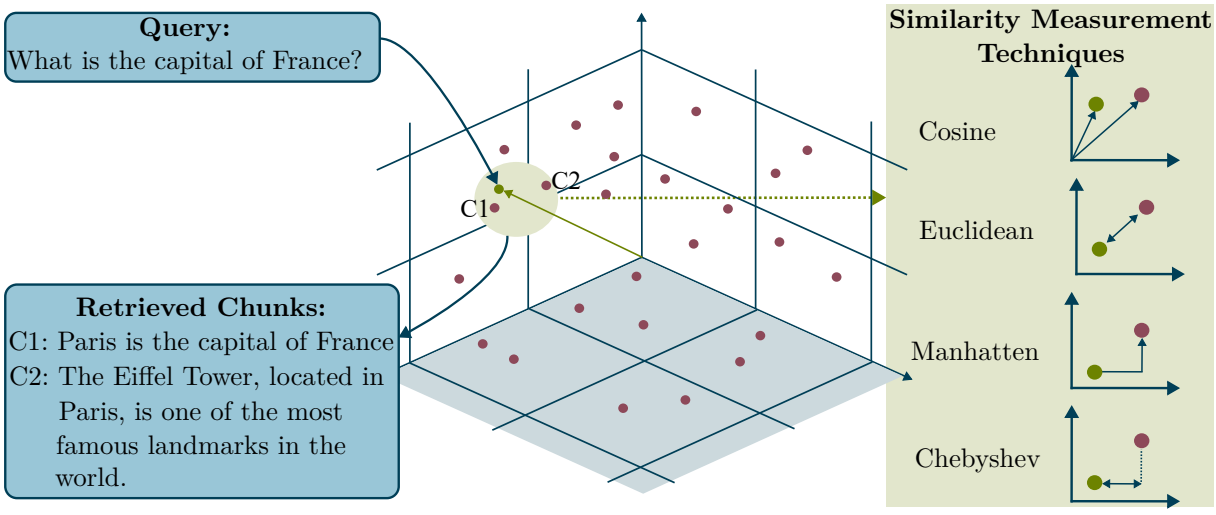

Figure 2.11.: Visualization of vector-based similarity search with different distance metrics

This architecture anchors text generation in accessible and verifiable sources instead of depending exclusively on the statistical patterns learned during training. For aircraft system development, such grounding is essential: outputs must not only be linguistically coherent but also factually reliable, up-to-date, and fully traceable.

Evaluation of RAG Outputs

As just described, a RAG system consists of a *retrieval component* that selects the most relevant text passages from the vector database, and a *generation component*, which formulates an answer based on this material. The evaluation of a RAG pipeline must therefore consider both the quality of document selection and the factual accuracy and usefulness of the generated response, as well as the interaction of both steps [136, p.2].

For the retrieval component, classical metrics from the field of information retrieval are well established. According to Yu et al. [136, p.9], these include in particular:

- Precision@k measures the proportion of relevant chunks among the top-k retrieved results. If five chunks are returned and three contain the requested information, precision is 60%.
- Recall@k indicates how many of the overall relevant chunks in the corpus appear in the top-k. If there are ten relevant chunks in total and seven appear in the top ten, Recall10 = 70%.
- Mean Reciprocal Rank (MRR) captures how early the first relevant chunk is retrieved. The earlier it appears, the higher the score.
- Normalized Discounted Cumulative Gain (nDCG) evaluates both the retrieval of relevant chunks and their order within the ranking, assigning greater weight to highly relevant items that appear earlier in the results list.

These measures enable a systematic evaluation of how effectively the generation component is supplied with relevant and well-structured contextual information.

For the generation component, classical metrics such as BLEU, ROUGE, F1, or Exact Match are often applied (see Section 2.4.2). These metrics primarily measure *surface-level overlap*[9], that is the degree of correspondence in word forms or sequences between a generated and a reference answer. However, two responses may be semantically equivalent even if they use different phrasing or synonyms. Such surface-based measures, therefore, do not reliably capture whether a response is factually correct and faithful to the retrieved evidence. According to Yu et al. [136, p.5], more content-oriented criteria have increasingly moved to the forefront:

- Faithfulness evaluates whether the answer is consistent with the retrieved sources.
- Answer relevance assesses whether the answer directly addresses the original question.
- Answer correctness checks the factual consistency of the answer against a known reference solution.
- Hallucination rate measures how often statements are generated that are not supported by the sources.

To operationalize these criteria, the method of *LLM-as-a-Judge* has gained prominence. In this approach, LLMs act as evaluators, rating answers according to dimensions such as faithfulness, relevance, or correctness. Zheng et al. [138] demonstrate that strong models such as GPT-4 achieve high agreement with human judgments in publicly available open benchmarks. At the same time, Dietz et al. [139] caution that methodological risks such as evaluation bias or circularity must be considered when deploying LLM judges. Overall, it becomes evident that the generation component cannot be adequately evaluated using

[9]Surface-level overlap refers to traditional lexical comparison metrics such as BLEU or ROUGE that assess similarity based on word or n-gram matches rather than underlying meaning [137, pp.2–3].

surface-level overlap measurements alone. A combination of semantically grounded metrics and LLM-based evaluation methods is required to achieve evaluations that are both meaningful and scalable.

Another central characteristic of RAG systems is their strong dependency on configuration parameters such as chunk size, the number of retrieved documents (k), or the choice of vector store. The performance of a RAG system can vary considerably depending on these settings, making systematic parameter variation indispensable for assessing the stability and sensitivity of the system's behavior. In AI research, this process is commonly referred to as *hyperparameter tuning* and forms an integral part of system optimization. Sensitivity analyses help to determine how robust and generalizable a configuration is, rather than relying on coincidental outcomes of specific parameter choices. Recent studies have adopted this perspective and begun to treat hyperparameter selection itself as an optimization problem. Fu et al. [140], for example, model hyperparameter tuning as an online bandit problem[10], dynamically adjusting parameters such as retrieval depth. Ammar et al. [142] show that design choices regarding chunking, vector stores, or reranking produce clear trade-offs between latency and accuracy[11] that become visible in evaluations. Barker et al. [143] extend this perspective to multi-objective optimization, balancing not only accuracy but also cost, latency, safety (hallucination risk), and alignment simultaneously.

For practical applications, this implies that a RAG system cannot be configured "once and for all" in a permanently optimal way. Instead, its use in technical and regulated domains requires continuous sensitivity analyses to identify suitable parameter configurations and adapt them to evolving requirements. Evaluation thus serves not merely as a tool for performance assessment, but as a central instrument for optimization and assurance of reliable system operation within the application context.

Advantages and Challenges of RAG

Unlike standalone LLMs, RAG separates reasoning from knowledge access and grounds responses in curated, external sources. This architectural principle yields several advantages for engineering contexts. By retrieving information directly from authoritative manuals and certified databases, RAG reduces hallucinations and factual errors, with empirical studies reporting significantly lower error rates for queries[12] [132, 145]. However, the performance strongly depends on the specific RAG implementation, including factors such as the quality of the indexed sources, retrieval strategy, and domain adaptation. In addition, knowledge bases can be continuously updated with new standards, service bulletins, or research findings, ensuring that the system reflects the most recent regulatory and technical developments. Finally, RAG provides transparency by citing retrieved passages as

[10] An online bandit problem denotes a repeated decision-making task under uncertainties: the system explores different options, observes their performance, and increasingly prefers those configurations that prove beneficial [141, p.1]. Applied to RAG, this means that hyperparameters such as the number of retrieved documents (k) are dynamically adjusted while the system simultaneously learns which settings yield the best results.

[11] *Latency* denotes the time delay between issuing a query and receiving the model's response, often measured in milliseconds or seconds. *Accuracy* refers to the degree to which model outputs correspond to correct or expected results, typically measured against human judgments or ground-truth data.

[12] The extent to which hallucinations are reduced depends heavily on the specific RAG setup, particularly on the quality and structure of the retrieved content ("garbage in, garbage out"). In one controlled study using structured CVs as context, the factual accuracy rose from 7% (no context) to 94% (with context in the form of a CV) [144]. However, no comparable evaluations are currently available for aerospace engineering.

evidence for generated outputs, enabling engineers to trace back design-relevant information to its sources and thereby strengthening trust and verifiability in safety-critical contexts.

These characteristics have already proven beneficial in industrial practice. For instance, interactive support systems retrieve targeted information from aircraft technical documentation, including manuals and records provided by manufacturers, maintenance organizations, and regulatory authorities, to assist technicians in troubleshooting [146]. Specialized RAG applications provide engineers with direct access to airworthiness regulations and certification guidance [147]. More advanced variants integrate knowledge graphs and digital twins to support design decisions with context-aware retrieval [148]. Airbus has also reported productive deployments in operational documentation, where pilots can query extensive Flight Crew Operating Manuals (FCOM) and obtain accurate responses in seconds [149]. These examples underline that RAG is not a replacement for LLMs but a complementary extension: while LLMs contribute linguistic flexibility and creative formulation, RAG ensures factual accuracy, timeliness, and traceability, making it particularly well-suited for the aircraft field.

Despite these demonstrable advantages, current RAG implementations share a common limitation: they are primarily designed for information retrieval through single-query interactions. This challenge makes them exceptionally effective in scenarios such as technical support or regulatory question-answer use cases, where a user seeks an answer to a specific question. However, the demands of aircraft system development extend far beyond the retrieval of isolated facts, requiring engineers to integrate information across multiple domains, identify interdependencies among hundreds of documents, and systematically consolidate evidence into decision-ready knowledge. Engineers need to navigate extensive legacy documentation, evaluate which solutions can be reused, and iteratively validate assumptions across multiple system levels. The challenge is therefore not simply to retrieve isolated facts, but to support an iterative and repetitive workflow in which information from diverse sources is continuously compared, checked, and updated. Concretely, this requires capabilities to:

- consolidate information across multiple technical domains and system levels,
- compare and evaluate alternative solutions based on distributed evidence,
- maintain consistency and traceability across iterative review cycles, and
- accelerate repetitive knowledge processing tasks that currently consume substantial engineering resources.

Beyond Single-Query RAG: Recursive Retrieval and Multi-Hop QA

The limitations of single-query RAG become particularly evident in the early phases of aircraft development. Here, engineers are not seeking isolated answers, but have to iteratively compare legacy solutions, evaluate alternatives, and consolidate distributed evidence across hundreds of documents. This process is inherently repetitive: requirements, functions, subsystems, components, and architectures are reviewed in multiple rounds, checked against regulatory documents, and revised as assumptions evolve. Current RAG systems, optimized for single interactions, cannot capture this iterative workflow. Instead, more advanced approaches are needed that extend the challenge toward multi-step and recursive retrieval.

Between 2022 and 2025, a large body of research explored exactly this problem: how to design RAG systems that go beyond one-shot queries and can support multi-step reasoning, iterative refinement, or structured exploration of document collections. To address

this challenge, several complementary strategies have been proposed in the literature, each extending the retrieval process differently:

- Multi-Hop Question Answering: breaking down complex queries into sub-questions, so-called hops, that are answered sequentially.
- Iterative RAG: combining generation and retrieval in repeated cycles, where uncertain answers trigger additional retrieval steps.
- Structured Approaches: organizing large document collections in hierarchical or graph-based structures to improve retrieval efficiency and support exploration.

These approaches share the recognition that a single retrieval step is insufficient for complex queries. While they primarily focus on improving the accuracy of individual answers, the methodology developed in this dissertation builds on the same recursive and iterative principles but applies them to a different objective: systematically exploring and consolidating domain knowledge to accelerate the document-intensive phases of aircraft system development.

The following literature analysis situates these extended RAG approaches, published up to April 2025, and positions the proposed methodology within this evolving research landscape.

1. Multi-Hop Question Answering

The first concept extends the "conventional" RAG principle by decomposing complex queries into sub-questions that are answered sequentially. As illustrated in Figure 2.12, this multi-hop reasoning enables the model to connect intermediate facts. Instead of retrieving once and generating a single response, the system alternates between retrieval and generation in several steps. For instance, to answer "*Where did the physicist who developed the theory of relativity study?*", it first identifies the physicist ("*Albert Einstein*") and then retrieves where he studied ("*ETH Zurich*"), combining both pieces of information into the final answer.

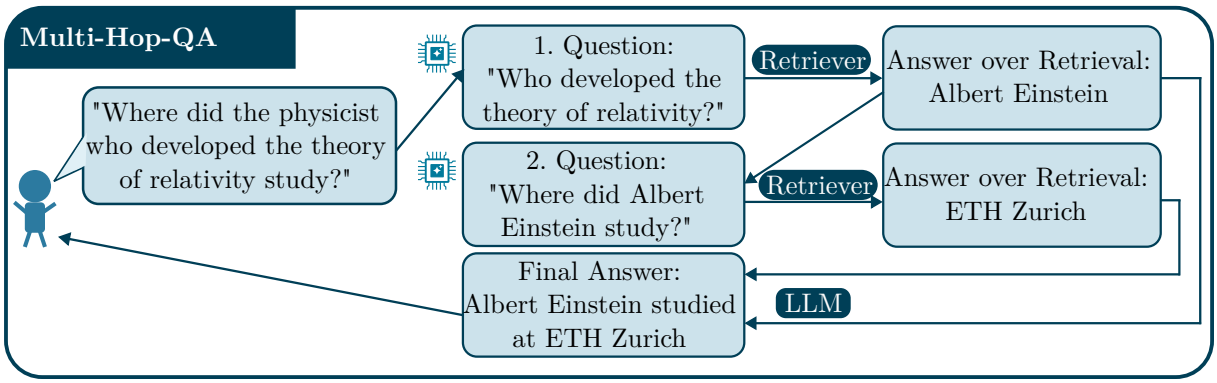

Figure 2.12.: Overview of the multi-hop QA concept

This principle was introduced by Press et al. [150], where an LLM autonomously generated follow-up questions. IRCoT by Trivedi et al. [151] combined this concept with Chain-of-Thought reasoning, embedding retrieval steps into explicit reasoning chains. CoRAG by Wang et al. [152] went further by fine-tuning LLMs on generated question chains to prepare them for systematic multi-hop reasoning.

The strength of these approaches lies in their ability to answer individual complex queries more accurately. In contrast, the methodology of this dissertation adopts recursion not to answer a single query, but to continuously expand the question space. Each answer

generates a new, more detailed sub-question, building an interconnected question-answer (QA) database that later supports structured synthesis.

2. Iterative RAG (Retrieval and Generation in Cycles)
The second research concept emphasizes iteration: initial answers are not treated as final results but are reviewed, refined, and expanded through successive retrieval cycles. This creates feedback loops between generation and retrieval that improve factual accuracy.

A simple example illustrates this principle. In response to the query "*When was the first moon landing?*", the system may initially retrieve documents mentioning that "*Apollo missions took place in the late 1960s*" and generate the answer "*1968*". In a subsequent round, a new retrieval step identifies a more precise passage stating that "*Apollo 11 successfully landed on the moon in July 1969*". Based on this, the system revises its response to the correct answer: "*1969*". An overview of the schematic concept can be seen in Figure 2.13.

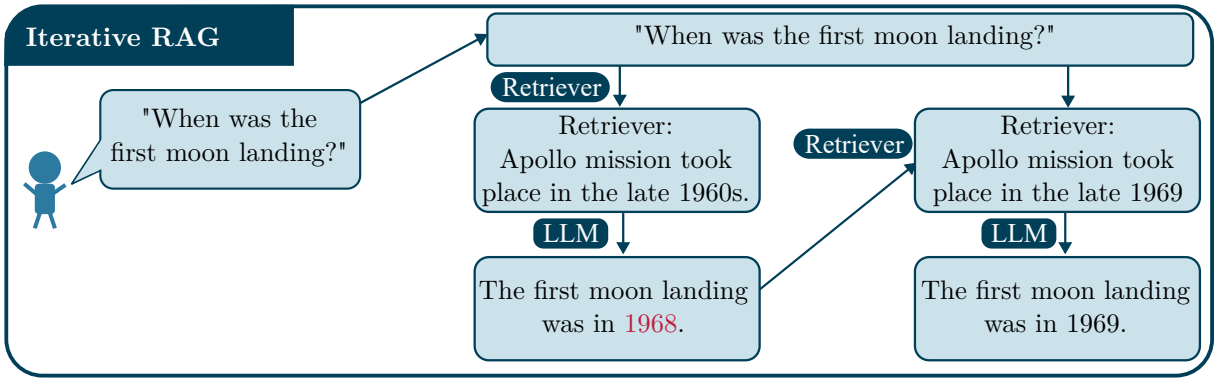

Figure 2.13.: Overview of the iterative concept

Different works have formalized this principle in concrete ways. One example is Iter-RetGen by Shao et al. [153] couples answer generation and new retrieval in an iterative cycle. FLARE by Jiang et al. [154] integrates an uncertainty estimate: while the LLM is producing text, it continuously checks its own confidence and only initiates retrieval when information is missing. Self-RAG by Asai et al. [155] extends this idea by incorporating self-reflection, in which the model evaluates the reliability of its own answer and triggers further searches if necessary. And DRAGIN by Su et al. [156] proposes a mechanism that dynamically determines information needs, ensuring that retrieval occurs only when truly required.

This method aims at maximizing factual accuracy in single responses. The methodology presented in this dissertation also employs iteration, but in a different way: recursive question chains are constructed not to refine a single answer, but to progressively cover a domain, with QA pairs stored and later synthesized into structured knowledge.

3. Structured Approach
The third concept emphasizes corpus preparation rather than query processing, addressing how knowledge is structured, indexed, and maintained. The goal is to organize large volumes of text so that the RAG system can access information more efficiently and in a hierarchical manner. An overview of the concept is presented in Figure 2.14.

A representative example is RAPTOR, introduced by Sarthi et al. [157], which recursively clusters and summarizes long documents into a hierarchical tree, from high-level summaries at the root to detailed passages at the leaves. In practice, long documents

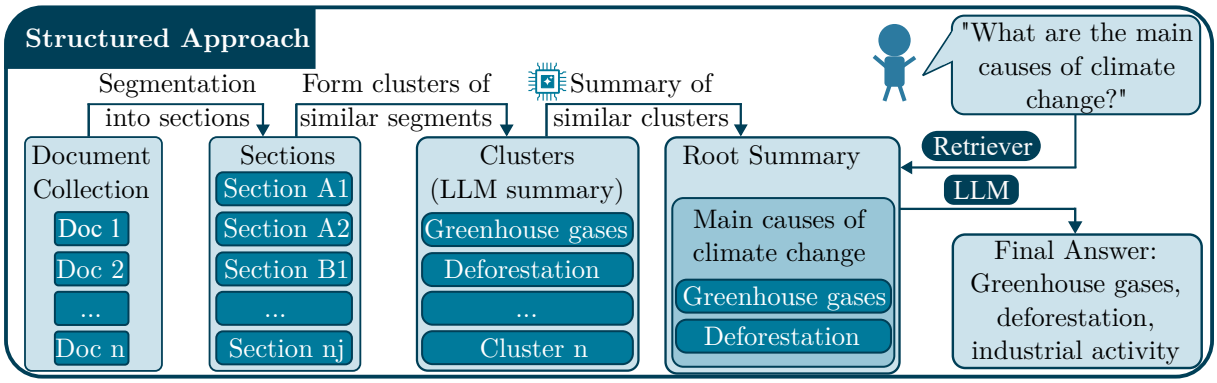

Figure 2.14.: Overview of the concept of structured approaches

are first segmented into smaller sections, which are then clustered and summarized using an LLM. These summaries form the next abstraction layer, which can be recursively clustered and outlined again. The result is a hierarchical tree structure: at the root, a very abstract overview (e.g., "*Main causes of climate change*"), at the intermediate levels, thematic clusters (e.g., "*Greenhouse gases*", "*Deforestation*", "*Industrial activity*"), and at the leaves, the detailed text passages. When answering a query such as "*What are the leading causes of climate change?*", the retriever can operate directly at the root level to provide a compact overview, while more specific questions trigger navigation into deeper branches of the tree, see Figure 2.14. Similarly, in GraphRAG [158] entities and relations from documents to construct knowledge graphs are extracted, which are clustered into communities and enriched with summaries. Other approaches, such as CRAG [159], focus on verifying the quality of retrieved passages and feeding back corrections when necessary.

All of these structured approaches share the idea of pre-organizing large text corpora into hierarchies or networks before the retrieval. Their main objective is to improve retrieval efficiency and precision. Similar structuring principles are also adopted in this dissertation, where an auxiliary LLM additionally summarizes domain-specific chunks to enhance retrieval coherence and contextual linkage. The detailed procedure for LLM-based chunk summarization and corpus structuring is described in Section 3.1.4. While sharing this general idea of structured organization, the SAGE methodology does not rely on static pre-structuring of the corpus. Instead, it applies similar principles dynamically, exploring the domain through recursive sub-questions and building a QA database with full source traceability, enabling systematic synthesis across the explored knowledge space.

While these approaches (multi-hop, iterative, and structured) mark significant methodological progress in the RAG research, they share a common focus: enhancing the factual accuracy and reasoning depth of individual question-answer interactions. However, their impact remains confined mainly to isolated tasks and benchmark settings. Despite these advances, the practical application of such extended RAG concepts in aircraft engineering is still in its infancy. The domain imposes unique challenges: engineers must navigate complex regulatory frameworks, trace design rationales across legacy systems, and integrate distributed evidence within the aircraft field.

The following section, therefore, reviews the emerging body of literature that applies LLM- and RAG-based methods directly to aircraft-related use cases. It focuses particularly on system and architecture design to assess how far current approaches meet the requirements of early-stage aircraft design and where methodological gaps remain.

2.4.4. Literature Analysis: RAG and LLMs in Aircraft Engineering

Building upon the general concepts of LLMs and RAG systems introduced in Sections 2.4.2 and 2.4.3, this literature analysis examines how both technologies have been applied to aircraft design and system architecture development. Because RAG and LLMs were originally developed for different purposes, the analysis distinguishes between two research streams:

1. RAG, which focuses on the retrieval, structuring, and contextualization of existing knowledge, and
2. LLMs, which emphasize generative and reasoning capabilities through in-context prompting[13].

While RAG excels at providing reliable and traceable access to existing data, it was not designed to generate new system knowledge. Conversely, LLMs demonstrate creative text-generation capabilities but lack mechanisms for consistent knowledge grounding. This conceptual distinction motivates the two-part structure of the following literature search: RAG-related studies are examined primarily in the context of aircraft applications, whereas LLM-related studies are analyzed with respect to system and architecture development.

To systematically capture existing literature, a structured keyword search was conducted in May 2025 across IEEE Xplore, Scopus, ACM, Web of Science, SpringerLink, and arXiv. The focus on *system design* reflects the scope of this dissertation: the early phases of aircraft engineering, where requirements, functions, and architectures are defined. These phases are particularly knowledge-intensive, relying on extensive document analysis, comparison of legacy solutions, and alignment with regulatory frameworks. While RAG and LLM applications have been explored in adjacent areas such as maintenance or operations, their role in systematic system and architecture design remains largely unexplored.

The literature search was divided into two complementary search spaces:

- Search Space 1 - RAG in technical domains: This search combined RAG-related terms (*retrieval augmented generation OR retrieval-augmented generation OR RAG*) with domain keywords (*aircraft OR aerospace OR aviation OR mechanical engineering*) to identify publications applying RAG in technical or engineering contexts, without restricting the scope to system design.
- Search Space 2 - LLMs in systems development: This search focused on studies employing LLMs (without mandatory retrieval components) directly in system or architecture development, including MBSE. The main keyword block (*large language model OR LLM*) was combined with domain and phase-specific terms (*aircraft OR aerospace OR airplane OR mechanical engineering*) and (*system architecture OR architecture design OR MBSE OR model-based OR systems engineering OR requirements OR design*).

The time filter was restricted to publications between 2020 and 2025, as the RAG concept was first introduced in 2020 [132]. Language settings were limited to English, and document types to journals, conference papers, and preprints. All search results were exported, deduplicated in Citavi, and subsequently screened according to PRISMA logic (title/abstract screening followed by full-text review). This dual search strategy enables a clear differentiation between retrieval-based and generative research streams within aircraft systems engineering.

[13] In-context prompting refers to the practice of embedding all relevant background information directly into the input prompt of an LLM. The model processes this temporary context during inference, but has no persistent memory or access to external knowledge sources.

RAG Applications in Aircraft Engineering

In the aircraft domain, RAG systems have primarily been applied as tools for structured information retrieval. Instead of generating new design concepts, these applications focus on giving engineers better access to relevant, validated, and traceable information from large technical databases.

Ren et al. [160] introduced the RAG-aided Causal Identification (RACI) model for aircraft accident analysis. By retrieving historical reports and generating causal summaries, RACI supports experts in tracing contributing factors and identifying causal chains in past incidents. Its scope remains analytical and retrospective, as the model is designed to interpret existing data. Similarly, Luo et al. [161] applied RAG to civil aviation policy queries, enabling more efficient access to regulatory knowledge. The system consolidates information from multiple regulatory sources. Xiang et al. [162] extended this principle to legal retrieval in aviation law, combining manual curation with automated retrieval to contextualize complex legal queries. In engineering contexts, Siddharth and Luo [163] employed RAG to extract design knowledge from patent descriptions, creating a structured knowledge base of existing design principles and configurations. Although these approaches differ in scope, they share a common goal: organizing and reusing information rather than creating new solutions.

More interactive applications have also emerged. The LeRAAT framework by Schlichting et al. [164] integrates RAG with real-time flight simulation data to generate context-aware recommendations for pilots. Here, the system retrieves relevant procedural guidance from manuals during flight emergencies, ensuring that the generated advice remains grounded in certified documentation. Similarly, Pandey et al. [165] presented OpenFOAMGPT, a RAG-enhanced assistant for Computational Fluid Dynamics (CFD). By retrieving tutorials and workflows, it improves accessibility to existing simulation knowledge but remains bound to predefined workflows. A broader comparative study by Silva et al. [166] systematically evaluated RAG and in-context prompting approaches for aircraft design question answering using the AeroEngQA benchmark. Silva et al. found that RAG performs similarly to in-context prompting and, in some multi-hop reasoning tasks, even slightly better, likely because the LLM in the RAG setup operates on a narrower and more specific information basis, rather than having to filter relevant facts from a large and unstructured prompt. Finally, Zhang et al. [167] developed a domain-specific RAG system for aviation safety that combines real-time data, accident reports, and fault logs to improve query precision and contextual accuracy.

In summary, RAG systems in the aircraft field serve as enablers for structured, evidence-based information access. They enhance transparency, traceability, and reusability of existing knowledge. Their methodological strength lies in the reliable retrieval and contextualization of validated information, rather than in creative synthesis. In contrast, LLM-based approaches shift the focus from knowledge retrieval to knowledge generation. They explore the potential of generative reasoning for architecture synthesis and design support, yet operate without persistent knowledge bases or retrieval mechanisms. They employ in-context prompting, where all potentially relevant background information is embedded directly into the model prompt. The model then generates outputs by reasoning over this temporary context, without persistent access to external knowledge sources. The following section, therefore, reviews LLM-oriented approaches in aircraft system and architecture development, highlighting their creative potential.

LLM-based Architecture Approaches

A systematic literature search identified 18 scientific works on LLMs in aircraft contexts. The majority address adjacent areas such as predictive maintenance or trajectory pre-

diction. Only four studies directly address system architecture generation. Across these works, a recurring pattern emerges: LLMs demonstrate creative generative potential but face systematic limitations in knowledge anchoring and domain integration.

Alarcia et al. [168] embedded Object-Process Methodology (OPM) system models into LLM prompts, chunked into sequences of approximately 9600 tokens, and applied them to three test space missions. Their work demonstrates that structured MBSE models can be effectively combined with LLM generation, resulting in clearer and less ambiguous specifications. However, the evaluation relied primarily on subjective assessment and did not establish mechanisms for systematic knowledge management across multiple sources, which limits scalability for larger development programs. Timperley et al. [169] integrated LLMs with the MBSE tool Capella in the framework "NalMage", enabling automated spacecraft system architecture generation directly from textual requirements. Their case studies, including an ESA mission concept and a CubeSat[14] payload development, illustrate both the creative generative potential of LLMs and their compatibility with established MBSE environments.

Other works apply LLMs directly via one-shot or few-shot prompting. Gomez et al. [171] explored one-shot and few-shot prompting for the automatic generation of aircraft system architectures and CAD models. Their studies on hydraulic aircraft systems and turbine components show that LLMs can deliver useful outputs even without structured retrieval or MBSE integration. Nevertheless, the reliance on manual iteration and the absence of systematic cross-domain knowledge anchoring limit the approach's reproducibility and scalability for complex design tasks. While Krus [35] applied ChatGPT with predefined prompt templates and micro-templates to generate UML diagrams and Python configuration rules for hybrid propulsion systems. The paper highlights the feasibility of iterative refinement through prompt engineering but also demonstrates the limitations of ad-hoc prompting: the lack of persistent knowledge bases and systematic validation mechanisms makes the approach fragile for comprehensive system architecture development. Moreover, the generated UML diagrams were not always entirely consistent or complete, requiring manual corrections or interpretations to align with the intended system logic.

Overall, the reviewed studies demonstrate that LLMs can be successfully applied to structured engineering tasks, such as architecture generation or model synthesis, when provided with well-defined contextual inputs. Building on this foundation, the present dissertation adopts a complementary approach that integrates retrieval-based knowledge extraction with generative synthesis. The RAG technology is used to identify, extract, and structure domain knowledge from large technical document bases, while the LLM leverages this curated information to generate consistent documentation. This design enables efficient knowledge synthesis across extensive technical documentation and supports systematic reasoning in the early development of aircraft systems. Together, these insights from existing RAG and LLM research set the stage for a broader synthesis of findings and motivate the methodological foundation introduced in this work.

2.5. Summary

The reviewed literature highlights complementary but incomplete approaches to applying GenAI in aircraft engineering. RAG demonstrates clear strengths in retrieving, contextualizing, and structuring existing knowledge, offering transparent and traceable access to verified information. Its use in the aircraft field has proven valuable for tasks such as

[14] A CubeSat is a type of miniaturized satellite with standardized dimensions, typically built in units of 10×10×10 cm, that allows for low-cost access to space for educational, research, and commercial purposes [170, pp.1-2].

accident analysis, regulatory queries, and safety-related decision support, yet its methodological scope remains confined to the extraction and organization of existing data rather than the generation of new engineering knowledge.

LLM-based approaches, in contrast, emphasize generative reasoning and creative synthesis. In the aircraft domain, they have been applied to generate system architectures, UML or CAD artifacts, and have been coupled with MBSE tools. However, all of these studies rely on *in-context prompting*, where relevant information is manually or programmatically embedded directly into the model prompt. This configuration assumes that the required knowledge is already available in a structured form. As a result, while LLMs demonstrate impressive creativity, they do not address how domain-specific information can first be extracted, filtered, and prepared from large, unstructured document collections.

At the same time, Chapters 2.1 - 2.3 have shown that aircraft systems engineering today relies on two development methodologies: DBSE and MBSE. In practice, most companies operate in both modes simultaneously. DBSE remains dominant due to its long-standing use and widespread familiarity, while MBSE offers advantages in consistency and traceability but faces adoption barriers, including required training, tool integration, and governance structures. Independent of the chosen methodology, a recurring challenge arises in the transition from high-level to detailed design: engineers must systematically analyze how existing requirements, functions, and components from earlier programmes can be reused, adapted, or newly developed. This process is inherently repetitive, time-consuming, and error-prone, making it a challenge in early system development.

It is precisely at this interface between documentation and design that the methodological foundation of this dissertation is situated. The SAGE methodology builds on the complementary strengths of RAG and LLMs to accelerate early-phase aircraft system design. RAG is employed to systematically screen, extract, and structure information from legacy documents, requirements specifications, and certification evidence while maintaining the transparency required by regulators. RAG selectively extracts the information relevant to system understanding and design, enabling LLMs to operate on concise, curated knowledge rather than entire document collections. Supplying an LLM with complete document collections is also not feasible in practice, as they cannot be integrated into a single prompt due to the model's limited input capacity. Structured according to the RFLP logic of systems engineering, this information forms a traceable representation of existing systems that can be seamlessly reused in subsequent design steps. On this foundation, LLMs are used to generate coherent documentation and synthesize new architectural insights. Findings from the reviewed literature reinforce this configuration, demonstrating that LLMs achieve high-quality and technically consistent outputs when provided with structured and contextually rich input data. The ability of RAG-based workflows to maintain explicit links between retrieved evidence and generated outputs supports certification-relevant auditability. This aligns with the European Union Aviation Safety Agency's (EASA) requirement that systems developed with AI/ML methods must "*operate at least as safely as traditional systems developed using existing development assurance processes and safety assessment methodologies*" [172, p.30]. Together, this targeted interaction between RAG and LLMs, selective retrieval followed by generative synthesis, forms the methodological core of the SAGE framework.

To the best of the author's knowledge and based on the structured search (May 2025) reported in this chapter, no prior work has addressed the central challenge examined in this dissertation: how domain-specific knowledge can first be systematically identified, extracted, and filtered from large technical corpora and then transferred into an LLM environment for reuse, adaptation, or new-system analyses under context-window constraints. The SAGE methodology developed in this work bridges precisely this gap.

3. Development of the SAGE Methodology

Addressing RQ1, RQ2, and RQ4[1]: How can the single-query limitations of traditional RAG systems be systematically overcome to enable comprehensive engineering knowledge extraction from technical literature while maintaining complete source traceability? Is it possible to systematically synthesize fragmented technical information into a coherent engineering document that meets established systems engineering standards? And can a GenAI-based methodology systematically combine knowledge from existing systems with new requirements to generate technically coherent system architectures?

Building on the research gap identified in Chapter 2, this chapter describes the SAGE (*System Architecture through Generative Engineering*) methodology. SAGE is designed to accelerate the development of aircraft system design by reducing the time required for the repetitive and resource-intensive analysis of existing systems against new requirements.

This chapter addresses RQ1, RQ2, and RQ4 by combining RAG with LLM in a four-phase structured process. Each phase corresponds to a specific methodological requirement: establishing a reliable knowledge base, extending retrieval beyond single queries, consolidating fragmented results into a coherent engineering document, and finally exploring systematic recombination for new system architectures.

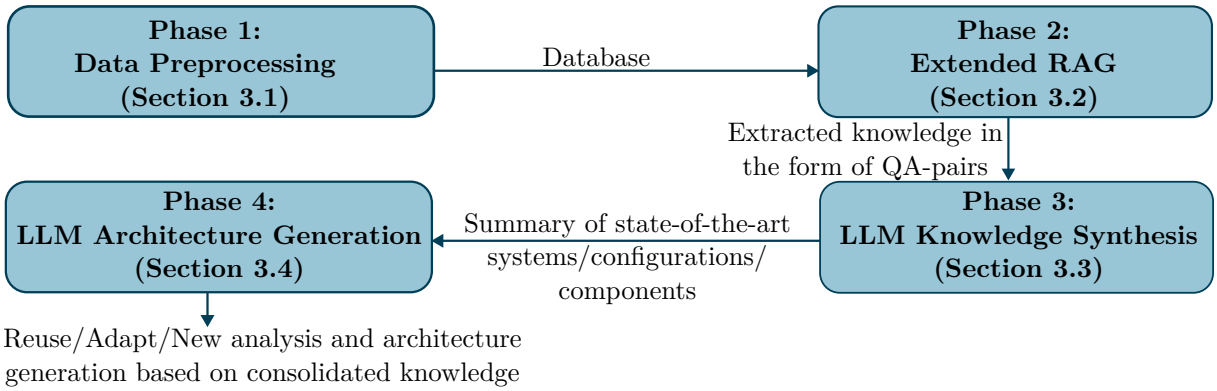

Figure 3.1.: Overview of the different phases in the SAGE methodology

As already mentioned in Chapter 1, the SAGE methodology consists of four integrated technical phases, as can be seen in Figure 3.1:

1. Phase 1: Data Preprocessing
   The first phase converts technical documentation into a structured and searchable knowledge base through text extraction techniques that extract textual information from scientific publications and patents. This phase employs parsing, semantic chunking, and vectorization to create searchable databases.

[1]RQ3 will be examined in Chapter 4

2. Phase 2: Extended RAG with Recursive Knowledge Extraction
   The second phase overcomes the single-query limitation of traditional RAG systems by automating hierarchical question generation, enabling the systematic exploration of large amounts of documents. Starting from systematically designed initial questions, the extended RAG system implements recursive question generation that automatically creates follow-up questions based on retrieved answers. This recursive process replaces conventional single-query approaches with hierarchically structured question chains, achieving exponential knowledge expansion across multiple levels of technical depth. It generates a comprehensive question-answer database while maintaining complete source traceability throughout the extraction process.

3. Phase 3: LLM Knowledge Synthesis
   The third phase transforms the fragmented question-answer pairs (QA-pairs) into a coherent engineering document by using an LLM. This systematic consolidation produces a structured technical report that follows the established RFLP framework, employs modular prompt strategies, and implements a three-tier confidence classification system to ensure the traceability and reliability of synthesized information.

4. Phase 4: LLM Architecture Generation
   In the fourth phase, an LLM derives new system architectures by recombining the synthesized findings into coherent design candidates. The process explicitly assesses subsystems and components for reuse, adaptation, or new development and documents the rationale for each choice. Guided by targeted prompts and reasoning LLMs, established engineering principles are applied to translate new requirements and predefined system boundaries into RFLP-consistent architectures based on the synthesized knowledge of existing systems.

By automating knowledge extraction, the methodology replaces time-consuming manual research and accelerates architectural analysis. The following section provides a detailed description of the implementation of Phase 1.

3.1. Phase 1: Data Preprocessing

The effectiveness of any RAG-based methodology fundamentally depends on the quality and accessibility of the underlying knowledge base. This phase transforms technical documentation into systematically accessible knowledge representations that enable comprehensive retrieval and analysis. The data preprocessing *pipeline*[2] addresses the challenge of converting heterogeneous technical literature into structured, searchable formats while maintaining complete source traceability and semantic coherence. All single steps of this phase are implemented in Python, using modular scripts to enable automation, transparency, and reproducibility. The structure of this phase can be seen in Figure 3.2.

3.1.1. Data Collecting

The first step of Phase 1 is to identify and compile suitable information sources that provide substantial insight into the current state of the system under development. The

[2]In this dissertation, a *pipeline* refers to a structured workflow consisting of sequential processing stages. Each stage performs a defined operation, such as cleaning, segmentation, or embedding, so that the overall process converts documents into machine-readable knowledge representations.

optimal information foundation would comprise detailed requirements specifications, descriptions of system boundaries and interfaces, technical documentation, and bills of materials, as these sources contain directly applicable knowledge about the system. The primary consideration regarding this type of documentation is its classification as confidential corporate information. A category of particular relevance within the aircraft industry, which results in its non-public availability.

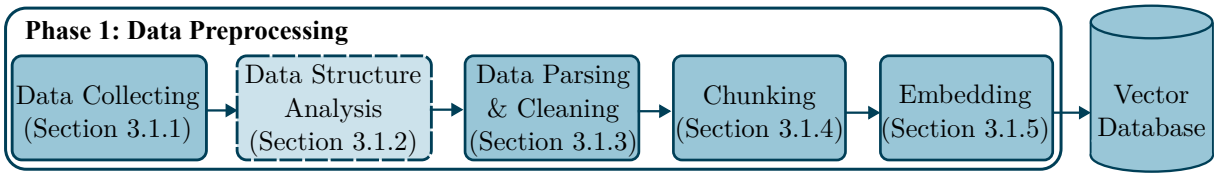

Figure 3.2.: Schematic structure of the data preprocessing pipeline, extended in this work with an explicit *Data Structure Analysis* step to reflect the characteristics of scientific publications and patents.

Due to these access restrictions, this dissertation adopts an alternative methodological approach, relying on publicly available knowledge sources, including scientific publications, regulatory documents, and patents. While these sources give a solid foundation for capturing domain-specific knowledge and current technological developments in system architecture, they are expected to provide less detail than internal corporate documentation, which remains inaccessible for this research. In the context of this dissertation, a distinction is made between *explicit* data, defined as structured and directly accessible information such as requirement lists, interface descriptions, or bills of materials, and *implicit* data, defined as knowledge embedded in unstructured textual sources, for example, within scientific publications or patent descriptions. Consequently, this dissertation analyzes only implicit data, as explicit data was not available due to confidentiality restrictions. In those sources, relevant information must be extracted from unstructured text and often interpreted between the lines, as design rationales or requirements are rarely stated explicitly but embedded within narrative or descriptive contexts.

The hybrid collection strategy described in the following subsections is primarily necessary when no prior data collection exists. If a validated dataset is already available, this step can be skipped, and the subsequent preprocessing stages can be executed directly.

Hybrid Collection Strategy

The data collection process follows a hybrid strategy that combines manual curation of patents and technical guidelines with automated acquisition of scientific publications. This mixed approach was necessary because not all data repositories offer open application programming interfaces (API) for automated access.

Repositories of patents and technical guidelines lack publicly documented APIs, and programmatic access was therefore not available to the author. Therefore, relevant documents were identified through a targeted manual search of patent databases and technical repositories. Each record was then reviewed to ensure its technical relevance and suitability for inclusion in the project database.

By contrast, many scientific publications were accessible via APIs that enabled structured and reproducible data acquisition. The automated retrieval process employed dedicated Python connector modules to query and extract publication metadata from sources such as the arXiv archive, Elsevier Scopus, and Springer Nature. The detailed implementation of this automated workflow is described in the following two subsections.

Programmatic Database Query and API Integration

As outlined, scientific publications were retrieved automatically through API interfaces, enabling a fully automated acquisition workflow. Dedicated Python connector modules handled the systematic querying and metadata extraction, replacing the need for manual interaction with the individual databases. In traditional literature reviews, keyword combinations and database queries must be constructed and executed manually, which quickly leads to a large number of individual searches. This makes the process time-consuming. To avoid this overhead, the present workflow automates both the generation of keyword combinations and the execution of all resulting queries.

Figure 3.3 illustrates the overall workflow. In the first step, several keyword groups are manually defined (for example, a primary system term, a secondary technology term, and optionally a contextual term such as the application domain). These keywords are then automatically combined into concrete search queries by joining one term from each group with the logical "`AND`" operator. This creates a complete list of all query combinations. The number and content of groups can be freely adapted.

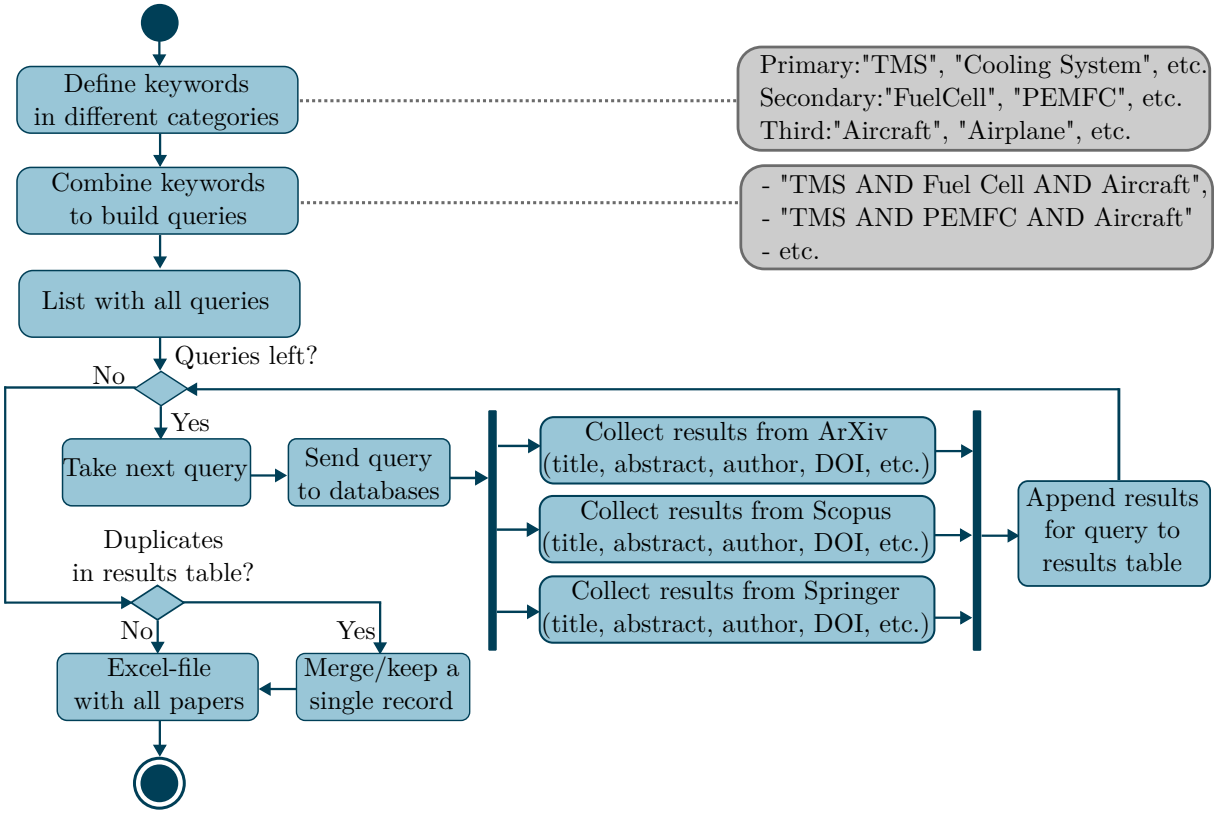

Figure 3.3.: Schematic activity diagram of keyword-based data acquisition

Each query string is processed sequentially. For every query, requests are sent in parallel to the selected databases (arXiv archive, Elsevier Scopus, and Springer Nature), while automatic rate limiting ensures compliance with provider restrictions. The returned metadata, like title, abstract, authors, source, year, and DOI, are continuously appended to a consolidated Excel file. With every new entry, the script checks whether the record already exists by DOI or, if unavailable, by title-year. Duplicate entries are therefore removed immediately during data acquisition, resulting in a continuously updated and cleaned document list that forms the input for the subsequent relevance assessment.

Multi-Dimensional Relevance Assessment Framework

After data acquisition, a relevance assessment framework evaluates the automatically extracted scientific documents based on their titles and abstracts. The framework employs analytical methods to establish an objective ranking of literature in the specific domain and research question.

The evaluation begins with a content-based analysis of titles and abstracts, as those are the only metadata that are systematically available through most database APIs. Each document is profiled according to domain-specific terminology using controlled vocabulary matching and NLP techniques. For example, to identify design-related papers, the Python script searches the titles and abstracts for terms such as "*design*", "*methodology*", or "*framework*". The focus on titles and abstracts reflects a practical constraint: in the databases used, only these elements are consistently accessible for automated processing via APIs. Although titles and abstracts are known to provide limited insight, particularly in aircraft publications, they represent the only consistent basis for large-scale automated relevance assessment.

The ranking combines multiple text-derived information, including the frequency and balance of domain- and methodology-related terms, document recency, and the absence of overly technology-specific terminology. These factors are aggregated using a weighted scoring function to generate a relative relevance score. This ranking allows the systematic prioritization of relevant literature on a large scale. Afterwards, the top-ranked documents are manually reviewed to verify content relevance and eliminate false positives. Finally, shortlisted publications are manually downloaded, as full-text access is unavailable through the database APIs.

3.1.2. Data Structure of Scientific Publications and Patents

The effective development of a RAG system for engineering documentation relies on the understanding of how different document types, particularly scientific publications and patents, are structured and presented. Those document types present particular methodological challenges due to their complicated organization and format-specific properties.

Structural Characteristics of Scientific Articles

Scientific publications in conference proceedings and journals adhere to an established structural convention that remains relatively consistent, despite variations between different publishers and disciplines. As shown in Figure 3.4, which displays a schematic excerpt of the first page of a journal publication, the typical structure begins with a concise title and an abstract that presents the essential findings of the work in a compressed form.

The introduction of scientific articles serves to contextually embed the research by outlining the research problem, explaining the relevance of the topic, and defining the specific research objectives. This is typically followed by a section on related research, where a critical examination of existing works takes place, and the research contribution is positioned. The methodological part of scientific articles provides a detailed description of the research design and the methods employed. In publications on system architectures, formal descriptions of algorithms, system models, or experimental setups are frequently found. This section exhibits high information density, combining various forms of representation. The results presentation is often multimodal, combining textual descriptions, tabular presentations of quantitative data, and visual presentations in the form of diagrams and graphics. These sections have a particularly high information density and pose

a challenge for automated information extraction procedures, as the semantic relationships between the different representation modalities have to be captured.

The medium of publication significantly influences the layout design of scientific articles. They often have a double-column layout. These formatting specifics make linear text extraction considerably more difficult, as the correct reading order must be reconstructed. Additionally, there are layout-specific elements, such as headers and footers, page numbers, marginal notes, and metadata, that must be distinguished from the actual content. Another characteristic feature of scientific articles is the extensive use of internal and external references. Internal references to figures, tables, equations, or other text sections establish a complex network of contextual relationships within the document. External references, in the form of literature citations, anchor the publication in the broader scientific discourse and provide valuable information about conceptual connections to other research works.

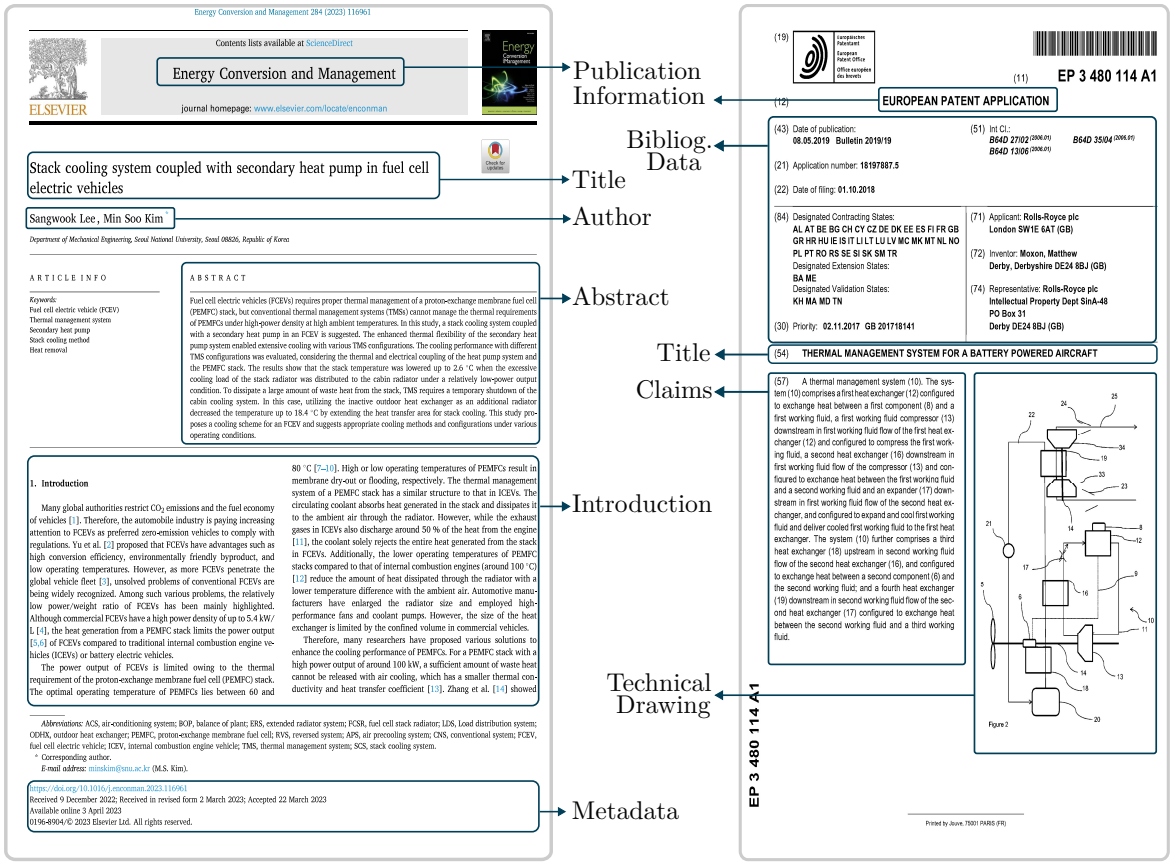

Figure 3.4.: Schematic overview of the structural characteristics of a scientific article (left) [173] and a patent document (right) [174]

Structural Characteristics of Patent Documents

Patent documents differ fundamentally from scientific articles in structure, language, and objective. While scientific articles primarily serve to communicate new findings to the research community, patent specifications serve a dual purpose: on the one hand, to disclose technical innovations for the benefit of society, and on the other hand, to secure legal protection for the applicant through patent rights.

The structure of patent documents follows an internationally standardized format, whose elements are shaped by legal requirements. As shown in Figure 3.4, patents begin with the bibliographic data at the top. These include essential metadata such as the publication date, application number, applicant, inventor, and international patent classification. The title of a patent specification, unlike that of scientific publications, is often deliberately formulated broadly to avoid unnecessarily restricting the potential scope of protection.

The description in patent documents typically begins with a presentation of the state of the art, in which existing solution approaches and their limitations are explained. Building on this, the technical problem to be solved is formulated, and the inventive solution is described in detail. A central element is the implementation examples, which represent concrete forms of the invention's realization and are often illustrated by technical drawings. This narrative-descriptive structure differs significantly from the analysis-oriented presentation in scientific articles.

The central part of the patent document is formed by the patent claims, which define the legal scope of protection that the patent provides. They follow a highly formalized language with specific syntactic conventions and are hierarchically structured into main and subclaims. A balancing act characterizes the formulation of claims: On the one hand, they have to be precise enough to capture the essence of the invention, and on the other hand, they should remain sufficiently abstract to ensure the broadest possible scope of protection.

Another characteristic of patent documents is the system of technical drawings with corresponding reference signs. These drawings adhere to conventions regarding their design and are typically schematic, lacking decorative elements. The reference signs used in the text establish an explicit link between textual descriptions and visual representations, which represent a valuable resource for automated information extraction.

The documentary quality of patent specifications varies considerably, especially with older documents. Many patents are available as scanned documents. Technical terms, unusual formulations, and the reference numbers in technical drawings pose particular challenges for text recognition. This quality variance must be taken into account when developing information extraction methods.

Implications for Information Extraction

The structural characteristics of scientific articles and patent documents define specific requirements for the subsequent parsing process. The semantic linking of multimodal content in scientific articles requires methods that go beyond simple text extraction, capturing relationships among text, equations, and images. Additionally, the layout complexity of scientific articles with multi-column layouts and nested structures necessitates advanced document analysis methods to reconstruct the logical reading order.

For patent documents, variability in document quality poses a central challenge. Robust mechanisms are required to ensure reliable text extraction even with suboptimal scan quality. The correct assignment of text references to elements in technical drawings through the reference numeral system requires specialized rule-based scripts that can integrate textual and visual information.

Consideration of these document-specific characteristics is fundamental for developing effective information extraction strategies in the context of an RAG system. Only through a deep understanding of the structural properties and content-specific characteristics of scientific articles and patent documents can parsing methods be developed that fully exploit the information potential of these sources and make them usable for the automated generation of new system architectures.

3.1.3. Data Parsing & Cleaning of Textual Document Elements

The processing of the collected literature involves a multi-stage process, as illustrated in Figure 3.5. While a comprehensive multimodal approach could theoretically extract text, mathematical formulas, and pictures to preserve the rich semantic content of technical publications, such an approach would inevitably lead to considerable complexity, significant computational effort, significantly longer processing times, and substantially higher operating costs, as special LLM functions would have to be used on a large scale for formula recognition and image analysis.

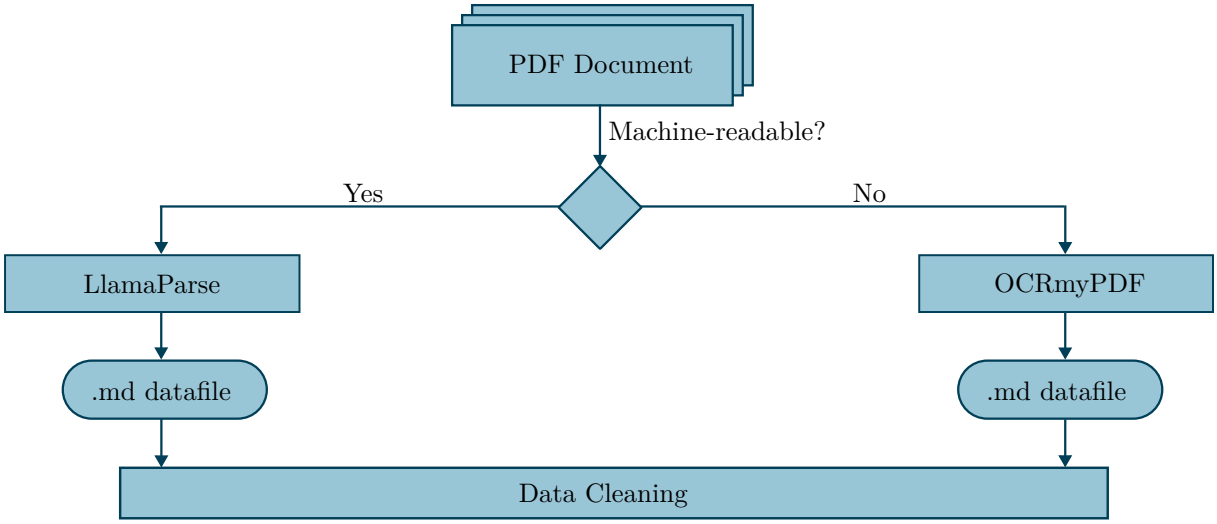

Figure 3.5.: Flow diagram of a PDF preprocessing pipeline that analyzes different document types and extracts, processes, and combines content into a common text data format using specific parsers

Initial experiments were conducted to evaluate the feasibility of incorporating visual elements into automatic image description. While the technical extraction of images from PDF documents proved straightforward, the automatic interpretation of image content, especially system architectures, posed a significant challenge for state-of-the-art LLMs. The descriptions generated by the LLM were occasionally correct but just as often incorrect, even when accompanied by explanatory text from the scientific paper for more context. System architecture diagrams without detailed annotations present numerous opportunities for misinterpretation by LLMs, as the models struggle to identify component relationships accurately and data flows without explicit guidance. A detailed description of the experimental image extraction and interpretation workflow, including representative examples and evaluation, is provided in Appendix A.4 on page 186. Future approaches may require standardized formats such as vector graphics (EPS) or systematically numbered system components and connections to enable reliable automatic interpretation. Therefore, this implementation deliberately focuses on the high-quality extraction and processing of text content, including mathematical formulas and table structures, while systematically excluding visual elements.

Data Parsing

A variety of specialized parsers are available, each offering distinct strengths and weaknesses for document processing. The evaluation and selection were conducted between

February and March 2025 to identify the most suitable parser for the use case. As parser technologies continue to evolve, future assessments may yield different results. Selecting the optimal parser is fundamental to the overall quality of the RAG system in Phase 2, since inaccuracies or structural deterioration at this early stage cannot be compensated by downstream process steps. The comprehensive parsing workflow implemented in this research is shown in Figure 3.5 and highlights the differentiated processing paths. Each path employs specialized parsers tailored to the respective document types and content elements.

The systematic evaluation of the different parsers was based on a comprehensive set of criteria that considered both quantitative aspects, such as completeness and processing speed, as well as qualitative characteristics, such as structure preservation and handling of special document elements. The heterogeneity of the underlying source material, comprising scientific publications and patent documents with varying structural properties, necessitated this differentiated approach to ensure optimal extraction quality. The criteria can be seen in Table 3.1.

Table 3.1.: Evaluation criteria for parser comparison

Criterion	**Description and rating scale**
Completeness	Comparison of the number of words to the original text.
Clarity	Structure and readability (i.e., paragraphs preserved?) *Scale: 1 = Not clear, 2 = Clear, 3 = Very clear*
Sentence interruptions	Unnatural sentence breaks caused by faulty line breaks. *Scale: 1 = Frequent, 2 = Sometimes, 3 = None*
Tables	Correct extraction of tables with format. *Scale: 1 = No formatting, 2 = Recurring errors, 3 = Perfect*
Special characters	Correct transfer of umlauts and unicode characters (%, °, etc.) *Scale: 1 = No, 2 = Partially, 3 = Always*
Document struct.	Retention of headings, paragraphs, lists, bold/italic formatting. *Scale: 1 = None, 2 = Recurring errors, 3 = Perfect*
Image captions	Correct preservation of image captions. *Scale: 1 = No, 2 = Yes, but with line breaks etc., 3 = Yes*
Math. equations	Correct reproduction of formulas. *Scale: 1 = Random characters, 2 = Poorly readable, 3 = Usable*
Column handling	Correct transformation of multi-column layout into readable text. *Scale: 1 = Order destroyed, 2 = Many errors, 3 = Occasional errors, 4 = Perfect*

To ensure an objective evaluation, two test documents were selected: a scientific paper comprising 16 pages, 28 images, and 8 tables, and a patent document with 18 pages and 7 images. The documents were processed through all parsers for evaluation, and the results were compared using the established criteria set. For the quantitative completeness analysis, three representative text passages of the scientific paper were analyzed: the abstract, introduction, and page 2. This selection allowed the direct comparison between OCR-based parsers (Optical Character Recognition, designed for scanned image-based PDFs) and machine-readable parsers (designed for text-based PDFs). Although OCR-based parsers are primarily intended for image-based documents, they can also process machine-readable PDFs. The completeness percentage was calculated by comparing the number of words extracted by each parser with the original reference word count in these

passages. The results of the evaluation of the various parsers for the scientific publication can be found in Table 3.2 and provide the following outcome:

- OCRmyPDF proved to be a solid solution with 98.9% completeness, medium values for document structure (2/3) and column handling (2/4), as well as good processing of image captions (3/3). However, the weakness in mathematical formulas (1/3) limited its applicability for technical-scientific documents.

- Pytesseract also achieved high completeness with 99.03% and showed medium values for document structure (2/3) and column handling (2/4). As one of the few parsers, it achieved an acceptable rating (2/3) for mathematical formulas. Weaknesses showed in occasional sentence interruptions and the inconsistent preservation of metadata. The OCR-based approach also leads to longer processing times for extensive documents.

- PDFMiner achieved high completeness with 98.9% and showed strengths in preserving document structure (3/3) as well as in column handling (4/4). However, serious deficits were evident in the processing of mathematical formulas (1/3).

- PDFPlumber showed acceptable completeness of 98.2%, but failed gravely in document structure preservation (1/3) and column handling (1/4). The library is not suitable because it cannot handle two-column layouts. Sentences are abruptly interrupted in the middle of the text flow, resulting in chaotic and unstructured text output.

- PyMuPDF showed acceptable completeness with 98.2%, but failed, like the PDFPlumber parser, gravely in preserving document structure (1/3) and column handling (1/4).

- PyPDF2 showed good basic functionality with a completeness value of 98.6% and effective column handling (4/4). However, the critical weaknesses were revealed in the high noise content due to excessive extraction of interfering elements such as tables and images without contextual embedding. The post-processing effort required to filter these elements was deemed disproportionately high, leading to the rejection of this option.

- LlamaParse with GPT-4.1 achieved the highest completeness with 99.8% and demonstrated high capabilities in preserving document structure (3/3), column handling (4/4), and table processing (3/3). It was the only parser to achieve a high rating (3/3) for processing mathematical formulas.

Notably, LlamaParse's high performance with GPT-4.1 surpassed that of an earlier evaluation with GPT-3.5 Turbo. The GPT-3.5 Turbo variant exhibited incomplete extractions, with missing text passages, as well as missing image and table captions. The switch to GPT-4.1 led to a significant quality improvement, especially in the contextual embedding of technical papers with the two-column layout.

For those patent documents that are exclusively available as scans, the evaluation focused on OCR-based solutions. The direct comparison between Tesseract OCR and OCRmyPDF showed a significant difference in processing speed with comparable result quality. Tesseract OCR required 11 minutes and 28 seconds for 7 PDFs with 84 pages, while OCRmyPDF processed the same test set in only 6 minutes and 52 seconds. This efficiency increase of 40% with consistent quality led to a clear decision for OCRmyPDF as the optimal solution for scanned patent documents.

Table 3.2.: Comparison of different parsers

Criterion	OCR-myPDF	Py-tesseract	PDF-Miner	PDF-Plumber	Py-MuPDF	Py-PDF2	Llama-Parse GPT-4.1
Completeness and Word Count							
Completeness (%)	98.89	99.03	98.89	98.23	98.23	98.61	99.8
Sum Words (orig.: 2050)	2073	2070	2073	2087	2087	2079	2054
Abstract (orig.: 282)	282	282	282	297	297	283	282
Introduction (orig.: 388)	390	390	393	392	392	393	388
Page 2 (orig.: 1380)	1401	1398	1398	1398	1398	1403	1384
Quality of Text Extraction							
Clarity	2	2	2	1	1	2	3
Sentence interruptions	3	2	2	1	1	3	3
Tables	2	2	1	1	2	2	3
Special characters	2	2	2	2	2	2	2
Document structure	2	2	3	1	1	2	3
Image captions	3	3	3	2	2	3	2
Math. formulas	1	2	1	1	1	1	3
Column handling	2	3	4	1	1	4	4

Implementation of a Hybrid Parsing Approach

The detailed evaluation leads to the development of a hybrid approach that combines the strengths of different parsers. For scanned patent documents, OCRmyPDF is used for processing because it is faster than Tesseract OCR and produces equivalent quality. This approach is particularly valuable for older patent documents that exist only in scanned format and require optical character recognition for content extraction.

For scientific publications and patent documents in machine-readable PDF format, LlamaParse is used throughout the process. After a systematic evaluation of various parsers, LlamaParse with GPT-4.1 demonstrated the most convincing performance in extracting complex scientific content, particularly for mathematical formulas, tables, and multi-column layouts. Unlike most open-source parsers such as PyMuPDF or PDFPlumber, LlamaParse is a commercial software that incurs usage-based costs. However, the higher quality in extracting and structuring complex content justifies this investment, especially since the limited number of documents to be processed keeps the overall effort within a reasonable range. The investment in extraction quality pays off through reduced post-processing effort and more precise information acquisition.

To further optimize the trade-off between extraction quality and costs, a two-stage hybrid parsing process was implemented. This design exploits the significantly lower cost of the GPT-4.1-mini variant while reserving the more expensive GPT-4.1 parser for selective use on mathematically complex pages. In practice, parsing with GPT-4.1 is 4.5 times more expensive than with GPT-4.1-mini. Apart from the handling of mathematical content, where GPT-4.1 provides superior extraction of equations in LaTeX format, both models deliver comparable quality in text and structural parsing. Although this pricing structure may change over time, the approach ensures both scalability and economic efficiency under current conditions. The implementation follows a two-stage process:

1. Initial Parsing with GPT-4.1-mini:
   The implemented solution uses LlamaParse together with the GPT-4.1-mini model in the first step. This parser is specifically optimized for efficient text content extraction and has been configured to ignore headers/footers deliberately. Simultaneously, it preserves tables in Markdown format while completely excluding images and graphics. The standard configuration already includes OCR support for mathematical formulas, providing basic recognition capabilities, although more complex formulas are not yet optimally processed at this stage.
2. Selective Processing with GPT-4.1:
   After the initial pass, a comprehensive mathematical content assessment is conducted on all pages using a sophisticated rule-based script. This script methodically identifies potential formulas by detecting mathematical symbols, equals signs, and structural patterns typical of mathematical expressions. When a page exceeds the empirically determined threshold of 4 potential formula elements, it is flagged for enhanced processing. These mathematically complex pages are then selectively reprocessed using a specially configured LlamaParse parser running with the GPT-4.1 model. This advanced parser is augmented with detailed instructions specifically targeting the precise extraction and proper LaTeX formatting of mathematical formulas, emphasizing the preservation of mathematical symbols and ensuring correct syntactical representation for both inline expressions and block equations.

This hybrid approach combines the cost efficiency of the standard parser for pages with predominantly textual content with the high quality of the GPT-4.1 parser for mathematically complex sections. The resulting output maintains the structural integrity of scientific documents with complex formulas, tables, and multi-column layouts while providing an optimal balance between extraction quality and economic efficiency. An example of the output after the parsing process is shown schematically in Figure 3.6 (middle panel).

Data Cleaning

After the parsing process, a systematic cleaning of the text is performed to optimize the quality of the data for the next phase. The transformation from parsed to cleaned text is visualized in the right panel of Figure 3.6. For this step, a rule-based script is implemented that fulfills the following tasks: removing DOIs, publication information (journal, conference, proceedings), and copyright notices, deleting page numbers and other numeric-only lines, filtering patent metadata (publication notices, sheet numbers, patent numbers), and removing repeated author mentions ($\geqq 3$), email or contact footers, as well as submission and acceptance notes. Non-content sections like references, acknowledgments, data availability statements, author contributions, funding information, and conflicts of interest statements are detected and deleted. The script also identifies the true content start (typically the abstract or introduction), deletes preliminary material such as author blocks, retains only the first occurrence of the title, removes lines dominated by low-information content and punctuation, and collapses excess whitespace. The result is a content-focused corpus that preserves the scientific body text (including tables) and improves downstream retrieval and vectorization by reducing noise.

This comprehensive cleaning process ensures that the knowledge base contains high-quality, content-focused text that maximizes the effectiveness of subsequent semantic analysis and vectorization processes while minimizing noise that could negatively impact retrieval performance.

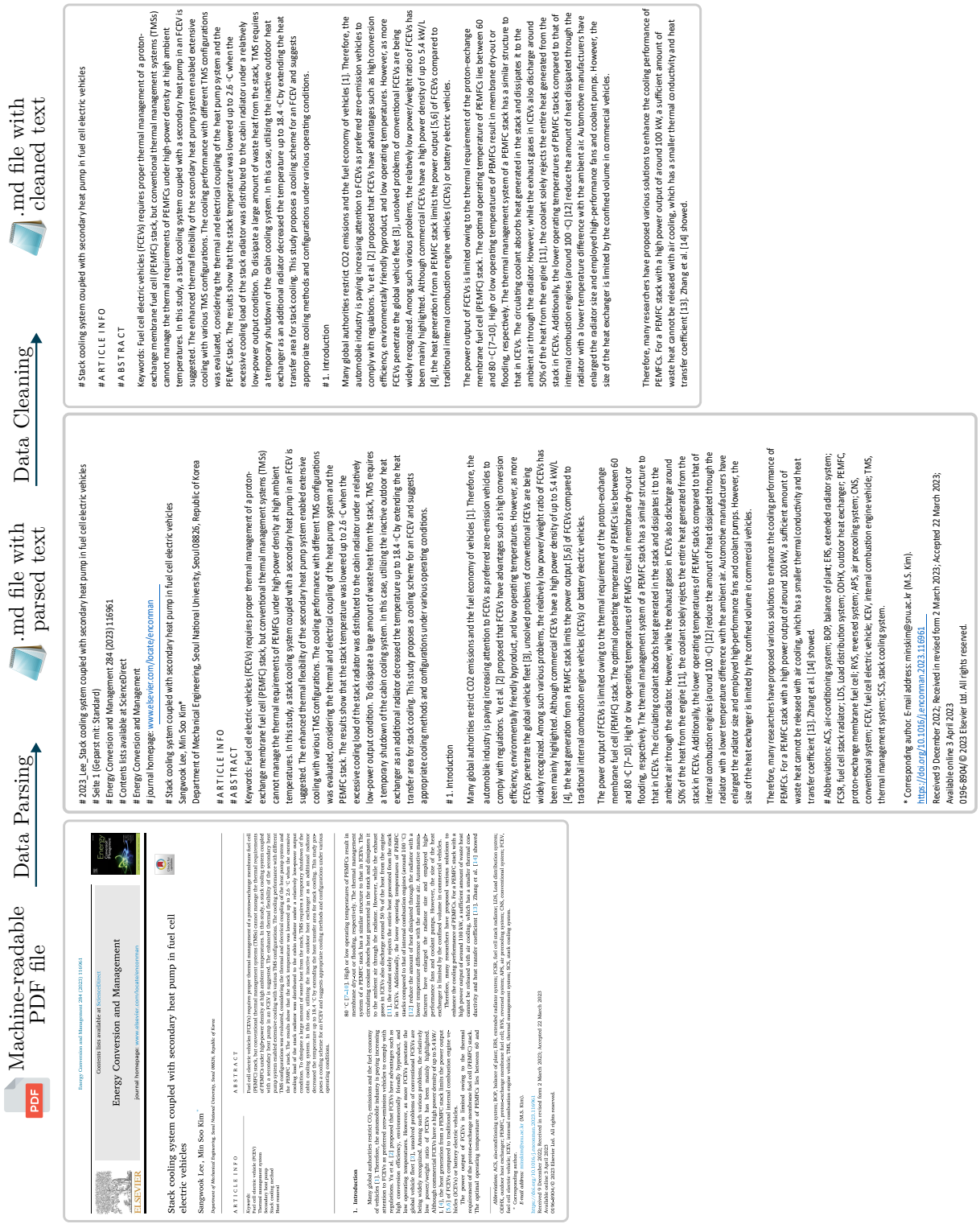

Figure 3.6.: Data preprocessing workflow for scientific publications using the first page of a fuel cell cooling system research paper by Lee and Kim [173]. Left: Original PDF file with layout-based formatting. Middle: Parsed text in Markdown format with preserved structure. Right: Cleaned text version after the cleaning process with optimized readability and removed formatting metadata. For better readability, the image has been rotated by 90°.

3.1.4. Data Chunking, Indexing, and Vectorization

The effectiveness of the RAG system also depends on the quality of its chunking and vectorization strategies. These process steps form the bridge between data preparation and information retrieval, determining how well the RAG can access and utilize the extracted knowledge. The following section describes the implemented approach to document chunking, semantic enrichment, and the creation of a vector database.

Semantic Chunking Strategy

Rather than using simple fixed-length chunking, which often breaks semantic coherence, this work implements a semantic chunking strategy. This approach divides documents into semantically coherent segments that preserve context and meaning.

The semantic boundary detection represents a significant advancement over traditional text segmentation methods. Rather than dividing documents at arbitrary character counts, the process employs NLP techniques to identify meaningful breakpoints, where content naturally transitions. The approach works by calculating embedding-based similarity between adjacent text segments using a sentence-transformer model, which captures the semantic relationships between different parts of the document. A drop in similarity score below a specified threshold marks a natural transition point, preserving idea coherence within each chunk while enabling segmentation for downstream retrieval tasks. This context-aware chunking preserves the flow of ideas within each segment and improves downstream retrieval quality.

Additionally, each chunk is automatically enriched with comprehensive metadata:

- chunk_id: A sequential integer ID assigned to each chunk
- node_id: MD5 hash of the content, a cryptographic algorithm that generates a unique 128-bit digital fingerprint of the text, enabling efficient integrity verification and duplicate detection across documents.
- file_name and file_path: Source document information
- creation_date and last_modified_date: Timestamps indicating when the chunk was created and last updated
- content_length: Character count for tracking chunk size
- LLM-enriched domain-specific categories (described in the following)

As an illustration of the implemented data structure, the example in Box 2 shows how a chunk appears. Each chunk is accompanied by metadata that captures both document information and LLM-extracted domain categories.

LLM-based Knowledge Extraction

As shown in the example in Box 2, every processed chunk is automatically annotated with domain-specific, LLM-enriched categories (e.g., “llm_SC”, “llm_RQ”). These categories are generated by a dedicated LLM-based extraction process that classifies each chunk based on domain-relevant engineering aspects. This automated categorization enables efficient retrieval based on specific system engineering aspects and systematic knowledge organization for subsequent synthesis processes. Conceptually, the process builds on the structured corpus preparation principles introduced in Section 2.4.3, which aim to enhance retrieval efficiency and contextual linkage in RAG systems.

Box 2: Representative Example of a Processed Chunk


```
{ "chunk_id": 46,
  "node_id": "2ddf1508-1d7b-4b33-a017-6793cfba7c33",
  "content": "# II. State of the Art # A. Fuel Cell Systems in ...",
  "metadata": {
    "file_path": "D:.",
    "file_name": "2023_Luebbe_Interconnections.md",
    "file_type": "text/markdown",
    "creation_date": "2025-05-12",
    "last_modified_date": "2025-05-12",
    "doc_id": "2023_Luebbe_Interconnections.md_49442_1e4d21dc",
    "content_length": "49442",
    "llm_SC": "[External systems: Aircraft systems, aircraft architecture,
               Environmental conditions: Aircraft flight conditions,
               Interfaces: Fuel cell stack, hydrogen and oxygen cond....]",
    "llm_RQ": "[Performance parameters: Maximum fuel cell energy...]",
    "llm_FU": "[Primary functions of the system: Convert chemical ...]",
    "llm_LE": "[Functional subsystems: Fuel cell stack, hydrogen ...]",
    "llm_PC": "[Components: Fuel cell stack, polymer membrane, ...]",
    "llm_AP": "[System architectures: Integration of FuCS into aircraft...]"
} }
```


The six categories used in this dissertation are derived from established SE principles, specifically building upon the RFLP framework (Requirements, Functions, Logical Architecture, Physical Architecture) as described in Section 2.1.1. To address the comprehensive knowledge requirements of aircraft system design, the traditional RFLP framework has been extended with two additional categories:

- System Context was added to capture the operational environment, external interfaces, and boundary conditions that influence the system under development
- Architecture Patterns was included to systematically capture proven design approaches, modular concepts, and redundancy strategies that represent valuable engineering knowledge

This extended RFLP framework (RFLP*) ensures comprehensive coverage of all technical system aspects relevant to the use cases discussed in this dissertation while maintaining compatibility with established SE methodologies. Throughout this work, RFLP* refers to this six-category framework. The categorization follows the systematic decomposition logic of the V-model, supporting structured knowledge organization for subsequent synthesis and architecture generation phases.

Each chunk undergoes systematic analysis through a specialized LLM prompt that examines the text content for information relevant to any of the six categories. When category-specific information is identified within a chunk, this information is extracted and stored as enriched metadata alongside the original chunk content, enabling both semantic similarity search and category-filtered retrieval in subsequent processing phases. The exact wording of the applied prompt, including category definitions and extraction instructions, is provided in Appendix A.5 on page 191. The six categories are defined as follows:

- System Context (SC): External systems, environmental conditions, interfaces, etc.
- Requirements (RQ): Performance parameters, safety requirements, etc.
- Functions (FU): Primary system functions, control functions, operating modes, etc.

- Logical Elements (LE): Functional subsystems, energy flows, operating states, etc.
- Physical Components (PC): Components, dimensioning criteria, connecting, etc.
- Architecture Patterns (AP): System architectures, redundancy approaches, etc.

While these six categories reflect the specific requirements of aircraft SE, the methodology's categorical framework is inherently adaptable. For different engineering domains or specialized applications, the categories can be systematically modified or extended to match domain-specific requirements and established engineering practices without altering the fundamental knowledge extraction and synthesis processes.

Vector Database Implementation

For efficient semantic retrieval, an embedding and indexing layer is implemented that transfers text chunks into a vector space and supports approximate nearest-neighbor search with metadata filtering. *Sentence embeddings* transform sentences or paragraphs into vector representations that place semantically similar texts close together in a high-dimensional vector space. They enable fast similarity search and clustering via simple vector operations (e.g., cosine similarity), which is essential for retrieval in RAG pipelines [175]. There are several established embedding models from various companies, including OpenAI, Google, Hugging Face, and Azure, among others [176]. For the implementation, the model (*sentence-transformers/all-MiniLM-L6-v2*) from Hugging Face was used. This model offers a balance between precision and resource efficiency. It produces a 384-dimensional dense vector space that keeps storage and search efficient, while preserving sufficient expressiveness for technical content.

The implementation utilizes ChromaDB as the underlying vector database, providing persistent storage of embeddings and associated metadata while enabling similarity search with good retrieval speed. The implementation offers flexible collection management for organizing different document categories and includes sophisticated duplicate detection through document and chunk identification mechanisms. This comprehensive approach to vector storage ensures both efficiency and reliability in knowledge retrieval operations.

3.2. Phase 2: Extended RAG with Recursive Knowledge Extraction

Addressing RQ1: How can the single-query limitations of traditional RAG systems be systematically overcome to enable comprehensive engineering knowledge extraction from technical literature while maintaining complete source traceability?

Building on the preprocessing pipeline described in Section 3.1, this phase introduces the core mechanism of the developed methodology: a recursive knowledge acquisition process that systematically overcomes the single-query limitation of traditional RAG systems. The goal is to transform the single-query retrieval into a structured, self-expanding exploration of the technical knowledge domain.

As illustrated in Figure 3.7, conventional RAG systems operate in a single, linear cycle. A user formulates a query, which is embedded, compared against a vector database, and used to retrieve semantically similar document chunks. These chunks are combined into a prompt that is passed to an LLM, which then generates an answer. This setup performs well for targeted information retrieval (e.g., in chatbot scenarios) but becomes inefficient for comprehensive knowledge extraction. Exploring an entire technical domain manually

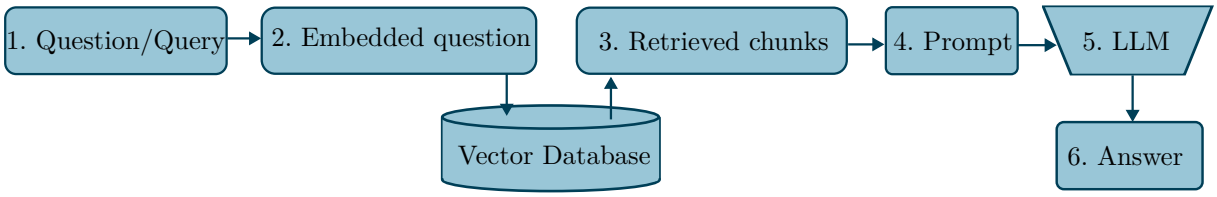

Figure 3.7.: The conventional retrieval process

would require hundreds of individual queries, risking inconsistent coverage, cognitive bias, and substantial effort.

This research addresses this limitation by introducing a recursive knowledge-acquisition mechanism that replaces isolated, manually formulated queries with an automated exploration process. Based on the answer to each question, the method generates follow-up questions that probe the topic at increasing levels of specificity, forming a controlled recursive cycle. While conceptually related to multi-hop and iterative RAG approaches (see Section 2.4.3), SAGE shifts the objective: instead of refining a single answer, the recursion is designed to achieve broad, coverage-oriented knowledge extraction. The process is guided by an SE-aligned category schema (RFLP*), ensuring that the expanding question hierarchy follows a structured systems-engineering logic. Each resulting question-answer pair is stored together with explicit source references, forming a traceable intermediate QA-corpus that serves as the knowledge foundation for Phases 3 and 4.

Figure 3.8 illustrates the resulting workflow:

- Retrieval and Answer Generation:
  Each initial question is embedded and used to retrieve semantically relevant chunks from the document database. The retrieved content and the question are combined into a prompt, from which the LLM generates a precise, source-linked answer. Each question-answer pair is stored in a structured QA database.
- Recursive Exploration via Follow-up Question Generation:
  Based on each generated QA-pair, an LLM formulates a set of follow-up questions that explore the topic at a deeper level. These questions are stored hierarchically and processed in the next recursion level, where the retrieval-answer cycle repeats. The process continues until a predefined recursion depth or question limit is reached.

The following subsections describe the conceived approach in detail with reference to Figure 3.8.

3.2.1. Categorized Initial Questions

The first step is the construction of a categorized question database. Instead of initiating knowledge capture through general or undifferentiated prompts, the approach applies a structured categorization aligned with the six engineering categories (RFLP*) defined in Section 3.1.4. This categorization forms the conceptual foundation of the questioning framework and ensures systematic coverage of all relevant system aspects.

For each category, a specific number of guiding questions were manually formulated to serve as the entry point for recursive knowledge acquisition. These questions were intentionally designed to be broad yet precise enough to capture meaningful technical information without constraining the scope of exploration. They provide a stable starting point for recursive expansion while allowing flexibility for adaptation to specific domains.

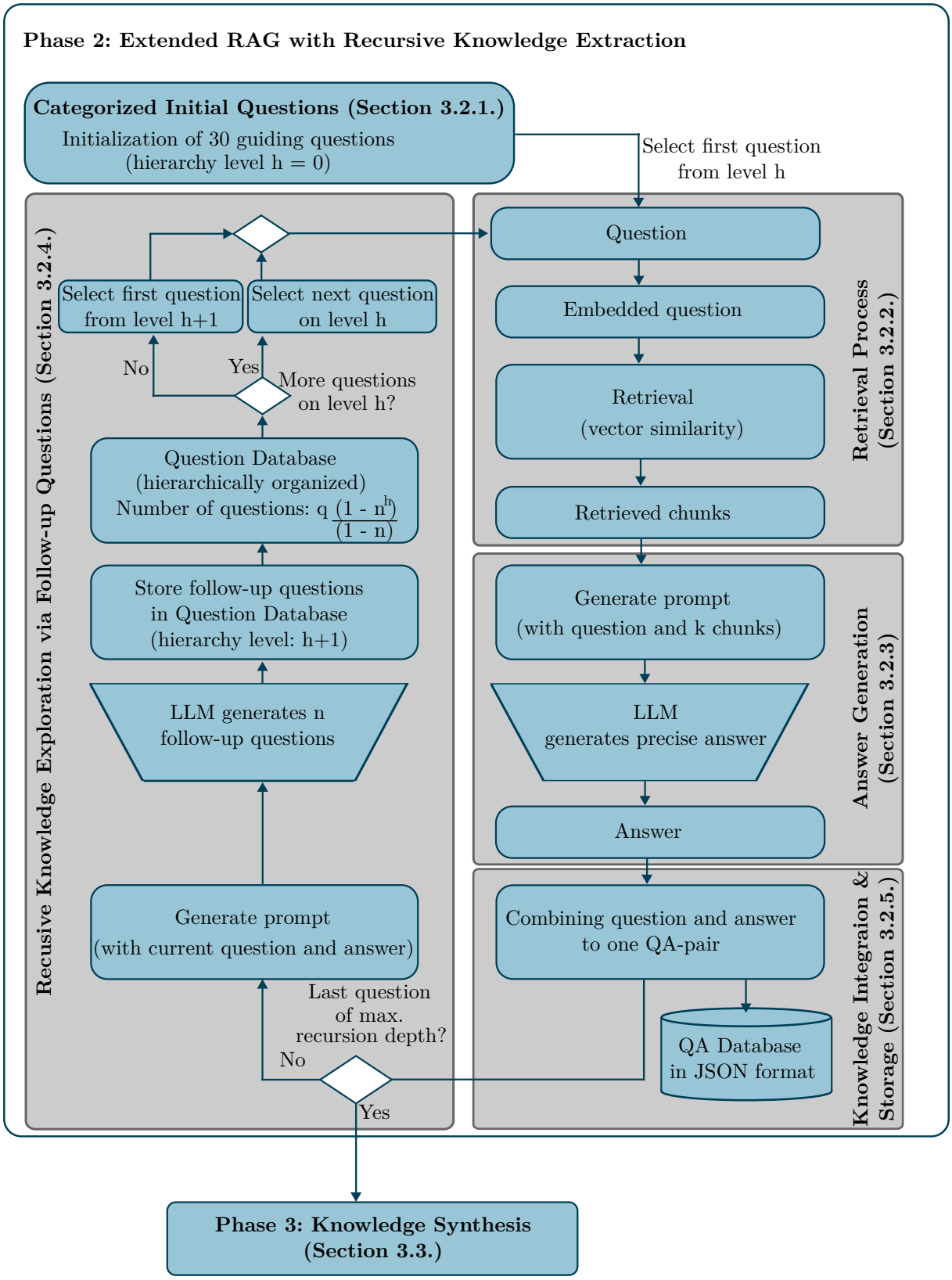

Figure 3.8.: Retrieval process flowchart: A comprehensive visualization of the two-path retrieval system, showing both the hierarchical question exploration path (left) and the direct answer generation path (right).

The categorization scheme is modular and extendable: additional categories (e.g., integration, verification, safety, or maintenance) can be added without altering the recursion mechanics. Likewise, existing categories may be subdivided into finer aspects (e.g., SC → stakeholders/environment/interfaces, RQ → performance/safety/regulatory) to increase analytical granularity when required.

The focus of this dissertation lies on developing and evaluating the recursive knowledge acquisition mechanism itself rather than optimizing the formulation of individual questions. While the initial questions and categories serve to demonstrate the method's functionality, the systematic design of optimal question sets represents a distinct opportunity for future research. The formulation of questions decisively shapes the scope and depth of extracted information, making question optimization a promising direction for further methodological refinement. In this dissertation, the approach employs 30 guiding questions (six categories × five questions), which serve as the foundation for demonstrating the effectiveness of the proposed methodology while acknowledging the potential for subsequent specialization and extension.

3.2.2. Retrieval Process

The retrieval process constitutes the second phase of the implemented RAG pipeline and identifies relevant information from the embedded vector database generated in Phase 1. The general concept of the retrieval process was introduced in Section 2.4.3. The following section describes the specific implementation used in this dissertation.

For each query, the system performs a semantic similarity search within the document vector database, as illustrated schematically in Figure 3.9. Both the query and the document chunks are represented as high-dimensional vectors, enabling cosine similarity-based comparisons to identify the most semantically related content. This vector-based approach enables the retrieval of conceptually relevant information even when different terminology or phrasing is used.

The document vectors also include the LLM-enriched category annotations introduced in Section 3.1.4. These additional descriptors increase semantic specificity and improve the retrieval of engineering-relevant content. Although the retrieval step itself remains a standard similarity search, the enriched vectors yield more relevant results for queries referencing requirements, functions, or architectural elements. After similarity scoring, the system applies a top-k selection to provide the most relevant chunks for the downstream synthesis steps.

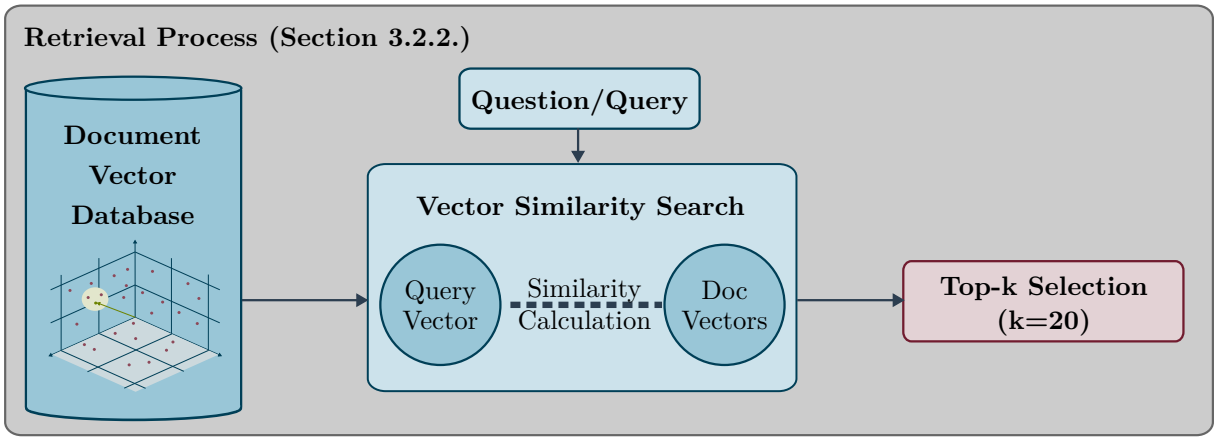

Figure 3.9.: Retrieval workflow implemented in this dissertation

3.2.3. Answer Generation

After retrieving the relevant chunks, they are integrated into a prompt template, as shown in Figure 3.8. This template combines the original question with the retrieved chunks and specific guidance for the language model. The constructed prompt is then passed to an LLM, which generates an answer based on the provided context. This approach ensures that the LLM response remains firmly grounded in the retrieved documents while maintaining natural language fluency. The prompt template follows a structured format, where {query_str} represents the original question and {context_str} contains retrieved chunks. For reproducibility, the exact wording of the prompt is provided in Box 3.

Box 3: Prompt Template for Answer Generation over LLM

```
You are an expert aircraft engineer specializing in {system_xyz} for aircraft.
Answer the following question about the aircraft system: {query_str}
Base your answer on the provided context information:{context_str}

Guidelines for your response:
1. Carefully analyze the user's question.
2. Examine the provided documents in the context to find relevant information.
3. Use Chain-of-Thought: Think step by step through your reasoning and document
                         your thought process.
4. Formulate a precise and direct answer to the question.

Important Rules:
- If the documents DO NOT contain sufficient information to answer the question,
  respond with: "No answer possible based on the provided documents."
- When using information from a document, you MUST include a reference like:
  [Ref: NODE_ID]
- Your answer must be supported by the provided documents.
- Avoid speculation or adding information not contained in the documents.

Format your response as follows:
- Thought Process: [Document your thinking process here. Analyze the documents and
  explain how you arrived at your answer or why no answer is possible.]
- Answer: [Precise answer to the question OR "No answer possible based on the
  provided documents."]
- Used-Sources: [Cite the sources that were used to create the answer. Use the
  following format: [NODE_ID]]
```

This prompt explicitly promotes the precise identification of relevant information and documents the thinking process (CoT). To prevent the model from hallucinating information, the temperature setting is set to a low value, which favors precise and factually correct answers. The *temperature* of an LLM controls the randomness of text generation. A low value (e.g., 0.2) makes the output more deterministic, while a higher value (e.g., 0.8) produces more diverse results [177]. According to Alammar and Grootendorst [178, p.127], "*the temperature controls the randomness or creativity of the text generated.*"

3.2.4. Recursive Knowledge Exploration via Follow-up Questions

The generation of follow-up questions is triggered only after an answer has been produced. At this point, both the question and its corresponding answer are used in a dedicated prompt (see Box 4) to formulate n targeted follow-up questions that further explore technical or contextual aspects of the topic. These are then assigned to the next hierarchy level and undergo the same process (retrieval, answer generation, follow-up question generation), forming a hierarchical question graph with explicit parent-child links.

Box 4: Prompt Template for Generating the Follow-up Questions over LLM

```
You are analyzing technical documentation about existing systems to extract
knowledge for developing new systems.
Based on this Q&A pair, generate {num_questions} follow-up questions that help
extract practical engineering knowledge.

Original Question: {question}
Answer: {answer}

Generate follow-up questions that:
- Extract specific design parameters, constraints, or requirements
- Identify relationships between components or subsystems
- Uncover design decisions and their rationale
- Explore implementation details and best practices
- Investigate performance characteristics and limitations

Format your response as:
1. [First follow-up question]
2. [Second follow-up question]
n. [{num_questions} follow-up question]
```

The recursion proceeds up to a defined number of levels. Consequently, the number of questions grows exponentially, as can be seen in Figure 3.10. If q is the number of initial guiding questions, n the branching factor (the number of follow-up questions generated per question), and h the number of hierarchy levels including the initial level, the total number of generated questions $N(h, q, n)$ is given by:

$$N(h, q, n) = \begin{cases} q \cdot \dfrac{1 - n^h}{1 - n}, & h \neq 1, n > 1 \\ q, & h = 1. \end{cases} \tag{3.1}$$

To illustrate this with the example shown in Figure 3.10, assume $q = 30$ initial guiding questions and $n = 2$ follow-up questions per level. If the recursion runs for $h = 4$ hierarchy levels, the total number of generated questions is $N(4, 30, 2) = 450$. The example

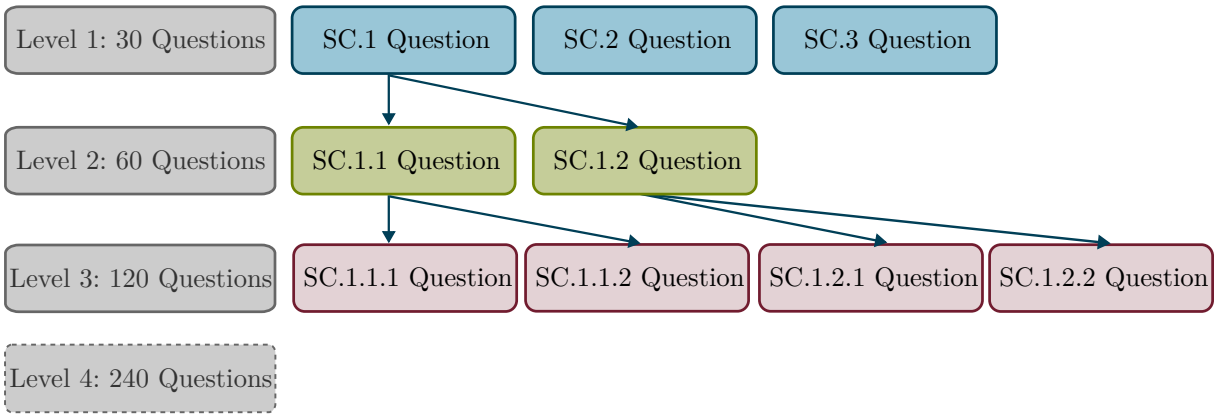

Figure 3.10.: Growth of question count across recursion levels with an assumption of 2 follow-up questions per guiding question and 30 guiding questions

visualized in Figure 3.10 illustrates this exponential expansion, where each node spans two successors until the defined recursion depth is reached. The follow-up questions aim to capture specific design parameters, component relationships, design decisions, implementation details, and performance characteristics.

3.2.5. Knowledge Integration and Storage

The generated QA-pairs are stored in a structured knowledge database using the JSON format, see Figure 3.8. This database employs a hierarchical organization comprising three essential components. First, metadata captures contextual information such as creation date, domain, and recursion depth, documenting the broader framework of the knowledge base. Second, the system organizes QA-pairs into the six predefined categories (RFLP*), preserving their hierarchical relationships to maintain the logical structure of the extracted knowledge. Finally, precise references to source documents accompany each information element, ensuring complete traceability and verifiability. This structured storage approach facilitates efficient management of the generated knowledge, enables targeted querying, and seamless integration into subsequent system design phases. The structural schema is shown in Box 5.

Box 5: JSON Structure of the Knowledge Database

```
{"metadata": {creation_date, domain, source_count, question_per_parent},
  "categories": {
    "<CAT (System Context, Requirements, etc.)>": {
      "questions": {
        "<ID>": {
          "question": "...",
          "answer": "...",
          "source_documents": [ ... ],
          "level": ...  }}}}}
```

3.3. Phase 3: LLM Knowledge Synthesis

Addressing RQ2: Is it possible to systematically synthesize fragmented technical information into a coherent engineering document that meets established systems engineering standards?

The recursive questioning process yields hundreds or thousands of QA-pairs that capture domain knowledge across diverse technical aspects and abstraction levels. However, these pairs represent fragmented information. To transform this fragmented knowledge into a coherent, assessable representation, Phase 3 leverages LLM synthesis capabilities to consolidate distributed content into a structured document aligned with engineering logic. This transformation enables engineers to evaluate and interpret the extracted knowledge more effectively. The process of this phase is illustrated in Figure 3.11.

As outlined in Chapter 1, a key objective of this phase is to investigate whether complex system descriptions can be systematically reconstructed from publicly accessible sources, such as scientific literature and patents. Accordingly, the synthesis stage integrates the QA-pairs generated in Phase 2 into coherent, technically consistent representations that mirror the hierarchical structure and functional logic of aircraft systems. In doing so, it transforms fragmented textual information into a structured, system-oriented representation that reflects the relationships and dependencies relevant to engineering design.

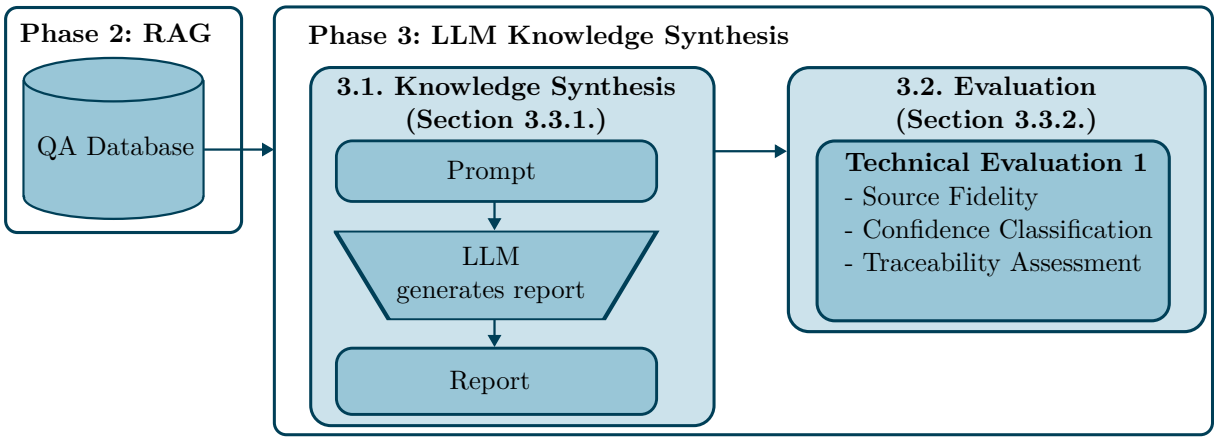

Figure 3.11.: Knowledge synthesis process

3.3.1. Knowledge Synthesis

A critical aspect of Phase 3 is the use of LLMs for synthesizing information while minimizing the effects of hallucination. To achieve this, the synthesis process applies a structured in-context prompting strategy that combines explicit task instructions with comprehensive contextual grounding. Each synthesis run is initiated through a dedicated prompt template containing two key elements: (i) detailed instructions specifying the synthesis objective, output structure, and reliability classification scheme, and (ii) the comprehensive set of QA-pairs obtained from the preceding recursive knowledge acquisition phase, which serves as the contextual knowledge base. Based on this structured input, the LLM generates an engineering report that consolidates distributed information in the form of the QA-pairs into a consistent, technically organized description.

By incorporating QA-pairs as contextual evidence, the model's reasoning is anchored in verifiable information. Nevertheless, the remaining risk of hallucination cannot be precisely quantified and must therefore be continuously monitored during evaluation.

Simultaneously with text generation, the LLM applies a three-tier confidence classification system, defined in the prompt, to enhance transparency regarding information reliability and enable risk-informed engineering decisions. This classification occurs concurrently with content synthesis rather than as a separate processing step, ensuring that reliability assessment is embedded throughout the document generation process:

- Direct Knowledge (Green): Information that can be explicitly traced to specific source documents and is stated with high confidence. This includes direct quotations, paraphrased content with clear source attribution, and information appearing consistently across multiple sources.
- Derived Knowledge (Orange): Information that represents logical inferences, generalizations, or interpretations based on the available sources, but which extends beyond what is explicitly stated. This includes relationships between components identified through analysis, performance parameters inferred from related systems, and functional characteristics deduced from architectural patterns.
- Hypothetical Knowledge (Red): Information that addresses gaps in the knowledge base through informed speculation, theoretical extensions, or hypotheses. This includes potential design alternatives not directly mentioned in the sources, anticipated performance characteristics without empirical validation, and possible system behaviors under conditions not explicitly documented.

Each statement in the generated report is linked to its originating QA-pair through explicit references. The integrated confidence classification enables engineers to quickly assess the reliability of different information elements and to prioritize evaluation efforts for areas with higher uncertainty. To ensure consistent visualization, the synthesized reports are produced in HTML format, allowing color-coded statements.

3.3.2. Evaluation of the Knowledge Synthesis Report

The knowledge synthesis report serves as a critical checkpoint in the methodology, requiring systematic evaluation before proceeding to architecture generation. This phase addresses the limitations of LLM-based approaches by incorporating human expertise and judgment while maintaining the efficiency advantages of GenAI-assisted processing.

While LLM evaluators, as described in Section 2.4.2 and 2.4.3, scale well to short-form answers, they are not suitable for evaluating the knowledge synthesis report due to its length and complexity. The decisive factor is the nature of the report: a long-form document with numerous source citations. Standard metrics such as BLEU, ROUGE, Exact Match, or n-gram F1 primarily capture word overlap with a single reference and miss semantically equivalent paraphrases. They neither measure chains of justification nor source grounding and correlate only weakly with human judgments on open-ended long-form outputs [119, 120, 136].

Therefore, the evaluation follows an engineering-judgment approach using self-defined criteria tailored to the SAGE methodology. Rather than linguistic similarity, the focus lies on the technical validity and source-grounded consistency of the synthesized document. Accordingly, three properties are assessed: source-faithful statements, interpretable confidence labeling, and end-to-end traceability across the RFLP* layers. Based on this, the following evaluation criteria are applied:

- *Source fidelity* checks whether synthesized statements remain faithful to the referenced material and do not drift through compression or paraphrase. In practice, the review concentrates on green-labeled statements (declared as directly supported), compares each claim to its linked QA-pair, and classifies the outcome as direct source alignment, value-added but misattributed (the fact exists elsewhere in the corpus but the link is wrong or missing), or source divergence (no support in the cited sources).
- *Confidence classification accuracy* evaluates whether the three-tier confidence system appropriately reflects the certainty levels of different information elements. The color-coded categories are reviewed to confirm clear differentiation between direct, derived, and hypothetical knowledge, with particular emphasis on orange (derived) and red (hypothetical) elements that require further expert validation.
- *Traceability* examines the internal logical consistency of the synthesized system representation across the RFLP* layers (Requirements $\rightarrow$ Functions $\rightarrow$ Logical Elements $\rightarrow$ Components). It evaluates the completeness and technical coherence of the relationships between these layers, i.e., whether each requirement is linked to a corresponding function, realized by defined logical elements, and ultimately implemented by physical components. Each mapping is classified as correct, incorrect, or missing. This provides a measure of the structural completeness of the synthesized knowledge base.

All evaluations were conducted manually in this dissertation to ensure precise assessment of LLM behavior and identification of potential inconsistencies. Reporting follows a deliberately simple structure: (i) counts and proportions of the statement-to-source matches for fidelity assessment, (ii) the distribution and manual evaluation of green/orange/red

confidence labels, and (iii) the number of correct, incorrect, and missing mappings for traceability. These checks support a clear go/no-go decision for Phase 4 and yield a focused revision list for improving synthesis quality.

Because LLM generation is inherently non-deterministic, all reported values refer to a specific synthesis run. However, the evaluation framework itself is reusable. Individual steps, such as claim-to-source comparison, can be automated or scaled up in future implementations without changing the underlying review logic.

This systematic evaluation ensures that the foundation for the subsequent phase is based on an accurate and comprehensive understanding of existing systems. Its iterative refinement capability allows continuous improvement of synthesis accuracy through expert feedback and the integration of additional information sources.

3.4. Phase 4: LLM Architecture Generation

Addressing RQ4: Can a GenAI-based methodology systematically combine knowledge from existing systems with new requirements to generate technically coherent system architectures?

Building upon the evaluated knowledge synthesis from Phase 3, this phase leverages LLMs to develop new system architectures that address new requirements or operate under different constraints. This process uses a systematic approach to apply established engineering principles. This enables the development of proven solutions through principle-based design and extends capabilities to new application domains. The process steps are shown in Figure 3.12.

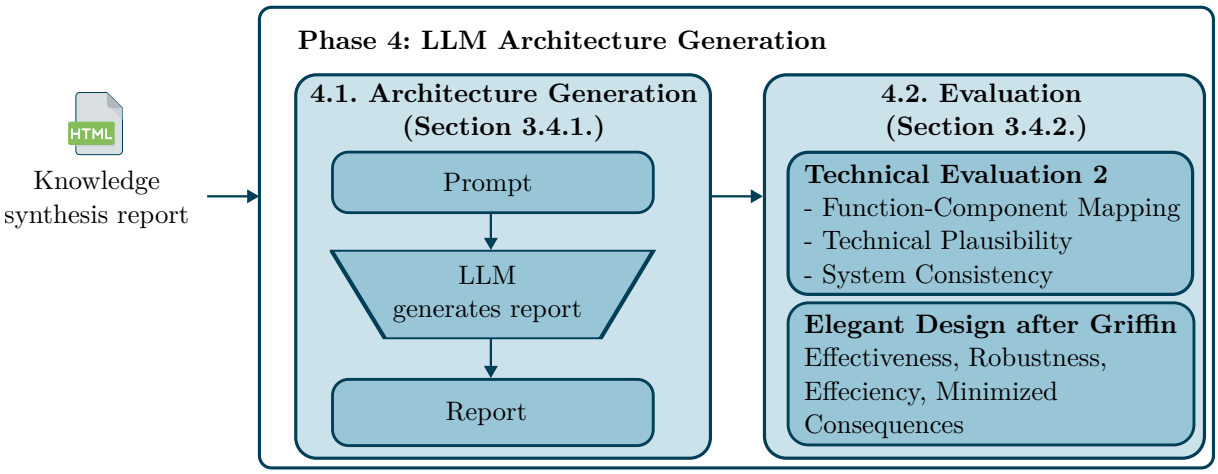

Figure 3.12.: Process of the system architecture generation process

3.4.1. Architecture Generation Process

The architecture generation process builds upon the consolidated knowledge synthesis report by translating structured information into new, system-specific architectures. This step marks the shift from knowledge aggregation to design reasoning, in which the LLM uses the insights it has retrieved and synthesized to propose functional structures, interfaces, and component configurations for the target system. Prompt engineering plays

a central role in this process because it defines how contextual information, design objectives, and output constraints are conveyed to the model. The general principles and techniques of prompting are summarized in Section 2.4.2 (page 27).

The implementation follows a lean, task-specific scheme with three mandatory elements: *context* (including assumptions, background, and the knowledge synthesis report from Phase 3), *task* (the requested operation), and *output format* (the required structure). The use of precise terminology and explicit formatting rules is important to ensure unambiguous system architecture content. A key design principle is the length of the prompt. Empirical observations in this dissertation indicate an inverse relation between prompt length and effectiveness. Very long prompts increase token cost and bury key instructions, which reduces compliance. Consequently, complex tasks are divided into concise, modular prompts with clearly defined sub-goals. This approach proved more reliable and effective than a single, comprehensive prompt. The architecture generation process itself unfolds in four consecutive stages, evolving from the reuse/adapt/new analysis toward a coherent system-level design.

- Stage 1: Component Reuse Assessment
  Identify which existing components can be reused, adapted, or must be newly developed. This step evaluates functional and physical compatibility, interfaces, and operating envelopes to determine the feasible starting point for system design.
- Stage 2: Context and RFLP* Reframing
  Establish the updated system context using the extended RFLP* scheme. Boundaries, interfaces, environmental conditions, and constraints are refined to derive the resulting deltas in requirements, functions, and logical structures.
- Stage 3: First Architecture Synthesis
  Combine the insights from Stages 1 and 2 into an initial, coherent system architecture. The result includes RFLP*-aligned specifications, rationale, expected performance, and implementation considerations, all supported by explicit evidence and confidence labels.
- Stage 4: Alternative and Trade Study
  Develop a second, technically distinct architecture option and compare it to the first. A structured trade study evaluates performance, complexity, cost, schedule, integration effort, and risk, concluding with a justified recommendation.

The final architecture report forms the foundation for detailed design, combining technical specifications with the rationale behind the chosen system architecture.

3.4.2. Evaluation of the LLM-developed Architecture

Following the comprehensive evaluation of the knowledge synthesis, the next phase evaluates the developed system design. While the synthesis evaluation focused on the accuracy of knowledge representation, the present step examines how effectively that knowledge has been transformed into an architecture that meets new requirements and constraints.

As in the knowledge-synthesis evaluation (Section 3.3.2), no established metrics exist for assessing LLM-generated system architectures. In this setting, standard LLM metrics such as BLEU, ROUGE, and Exact Match are not suitable for evaluation, as there is no unique reference architecture or text against which all outputs can be scored. Multiple architectures can satisfy the same requirements while differing in trade-offs. Consequently, word- or sentence-level similarity does not reflect requirement fulfillment, interface consistency, or compliance with engineering constraints. The evaluation criteria used here are therefore defined for this dissertation.

Technical-Systematic Evaluation builds upon the evaluation results of Phase 3 but shifts the focus from synthesized knowledge to the technical quality of the developed design. It assesses the methodology's performance across three analytical dimensions:

- Process Execution and Prompt Adherence: This criterion assesses how consistently the LLM followed explicit methodological instructions throughout all development stages. It evaluates adherence to prompt structure, completeness of deliverables, and compliance with format and process specifications. This includes verification that all required elements, such as the component reusability matrix, requirements specification, and comparative physical architectures, were successfully generated in accordance with the defined task structure.

- Mathematical Accuracy and Engineering Feasibility: This criterion assesses whether quantitative results and underlying calculations are accurate, reproducible, and technically sound. It assesses the accuracy of engineering equations, consistency of parameter use, and realism of numerical outcomes, considering current technological constraints and physical principles.

- Systemic Technical Understanding: This criterion examines whether the LLM demonstrates engineering reasoning and system-level understanding. It focuses on the ability to identify design flaws, propose corrective actions, and justify design decisions through transparent trade-off reasoning. Evaluation covers the logical consistency of system interactions, component integration, and the coherence of energy and material flows. High scores indicate that the model not only reproduces known relationships but actively engages in problem-solving and architectural reasoning.

Beyond these technical dimensions, the evaluation incorporates the **Elegant Design Framework** developed by Watson and Griffin [179]. This framework recognizes that effective SE transcends traditional approaches focused on individual components and their relationships. Instead, they emphasize "*context over structure, with interactions over elements, with the whole over the sum of the parts*". Watson et al. define elegant design through four interrelated attributes that capture how well a system achieves its intended purpose while operating harmoniously within its broader context [179, p.2].

The first attribute, *effectiveness*, evaluates how well the proposed solution meets all specified functional requirements, both explicit and implicit. The assessment considers the degree to which the developed system fulfills its primary functions under normal operating conditions, as well as its ability to perform secondary and supporting functions. Special attention is given to functions that are enhanced beyond the conventional system's capabilities, potentially opening new application domains or performance levels.

Robustness assesses the solution's ability to maintain performance under varying conditions, including unexpected inputs, environmental variations, and component degradation. The evaluation considers failure modes, redundancy mechanisms, and resilience strategies incorporated into the design. This attribute captures the extent to which the system resists significant deviations from expected behavior.

Efficiency compares resource utilization with the conventional system across multiple dimensions, including energy consumption, material usage, spatial requirements, manufacturing complexity, and maintenance needs. The evaluation quantifies efficiency improvements and assesses whether the architecture optimally utilizes resources throughout its lifecycle. This may include considerations of sustainability, recyclability, and total cost of ownership that extend beyond immediate operational efficiency.

The framework also examines the *minimization of unintended consequences* by analyzing potential integration challenges, operational issues, or secondary effects that could

negatively impact system performance or broader contexts. The assessment considers interactions with other systems, user interfaces, maintenance procedures, and environmental impacts.

This systematic evaluation enables a comprehensive assessment of both technical execution quality and innovation potential, providing objective criteria for comparing generated architectures with conventional systems and identifying areas that require further development or evaluation.

3.5. Discussion and Summary

This chapter developed the SAGE methodology to address the gaps identified in Chapter 1 and 2. The goal is to accelerate aircraft development by reducing the time required for repetitive analysis of reuse, adaptation, and new development options, while maintaining complete source traceability.

The methodology enables systematic exploration of knowledge spaces through categorized initialization and controlled recursion (Section 3.2). It produces a question-answer database with explicit source links and consolidates fragmented information into an RFLP*-structured document with confidence markings (Section 3.3). In Phase 4 (Section 3.4), the synthesized knowledge serves as the foundation for developing and evaluating new system architectures. Taken together, Phases 2-4 reduce manual literature work, increase coverage beyond individual expertise, and create a reusable, evidence-backed foundation for architecture generation and assessment.

But the methodology also has some limitations. The quality depends on the scope and fidelity of the underlying corpus and the model's capabilities. Public sources can be less detailed than internal documentation, which limits the depth of design. Generation is non-deterministic and requires review. The choice of parser, embedding models, and retrieval parameters influences cost and performance. Generated architectures must be evaluated before implementation. These constraints are mitigated through explicit source links, confidence classification, expert review, and the evaluation steps.

With respect to RQ1, RQ2, and RQ4, this chapter introduces methodological mechanisms intended to address these research questions rather than to claim definitive answers. For RQ1, the recursive retrieval process described in Section 3.2 extends conventional single-query RAG approaches by employing an SE-aligned seeding of questions and controlled follow-up generation. This enables comprehensive domain exploration based on verifiable evidence. For RQ2, the knowledge synthesis in Section 3.3 consolidates distributed QA-pairs into a structured engineering document aligned with RFLP* and enriched with confidence annotations. This provides the foundation for subsequent design reasoning and architecture generation. For RQ4, Phase 4 (Section 3.4) introduces systematic recombination mechanisms that build on the synthesized knowledge base to evaluate reuse, adaptation, and new development. This phase examines whether LLM-based synthesis can support the creation of technically coherent architectures, while also identifying instances where human engineering judgment remains indispensable.

While this chapter established the technical foundation for the 4-phase methodology, Chapters 4-6 describe the systematic optimization and practical application of this approach. Chapter 4 presents empirical parameter studies that analyze system performance across all four phases, while Chapters 5 and 6 demonstrate the practical evaluation of the methodology through two case studies: the Environmental Control System and the Thermal Management System.

4. Empirical Analysis of the Proposed Methodology

Addressing RQ3: What are the critical configuration parameters for deploying a GenAI-based methodology in aircraft engineering environments?

This chapter addresses RQ3 through the systematic empirical analysis of critical parameters within the SAGE methodology, as developed in Chapter 3. The methodology comprises several adjustable parameters across its operational phases. However, this chapter focuses on those with the most decisive influence on effectiveness, efficiency, and output quality.

In this dissertation, the term *parameter* refers to all variable configuration points within the methodology that can be adjusted to optimize system performance. These parameters represent the controllable aspects of the developed methodology that directly influence output quality and system behavior. In the sense of computer science, and following the NIST definition, a parameter is "*a value that is used to control the operation of a function or that is used by a function to compute one or more outputs*" [180]. This clarification is important, as the notion of parameter may differ in other disciplines, such as mechanical engineering, where it often denotes fixed material or design constants rather than configurable system inputs.

As illustrated in Figure 4.1, numerous parameters exist across all phases of the methodology. The following section systematically investigates the parameters that impact system behavior, including retrieval effectiveness, model performance, prompt architecture, and knowledge base composition, while acknowledging that other parameters were configured according to established practices and practical considerations. Following this analysis of the different system parameters across each phase, the parameters with a strong influence are examined in dedicated studies in Section 4.2, as indicated in Figure 4.1.

4.1. System Parameters: Classification and Influence Analysis

Following Chapter 2, parameterization is considered with respect to retrieval quality, generation quality, and their interaction. The methodology comprises four main phases, each with parameters that impact various aspects. The configuration points depicted in Figure 4.1 enable targeted adaptation of the methodology to specific use cases and domain requirements. In the following, the most essential parameters of each phase are systematically explained and linked to references in Chapter 3.

4.1.1. Data Preprocessing Parameters

The first phase, methodologically introduced in Section 3.1, forms the foundation for the quality of the entire methodology. The parameters significantly influence the form and availability of information for subsequent retrieval.

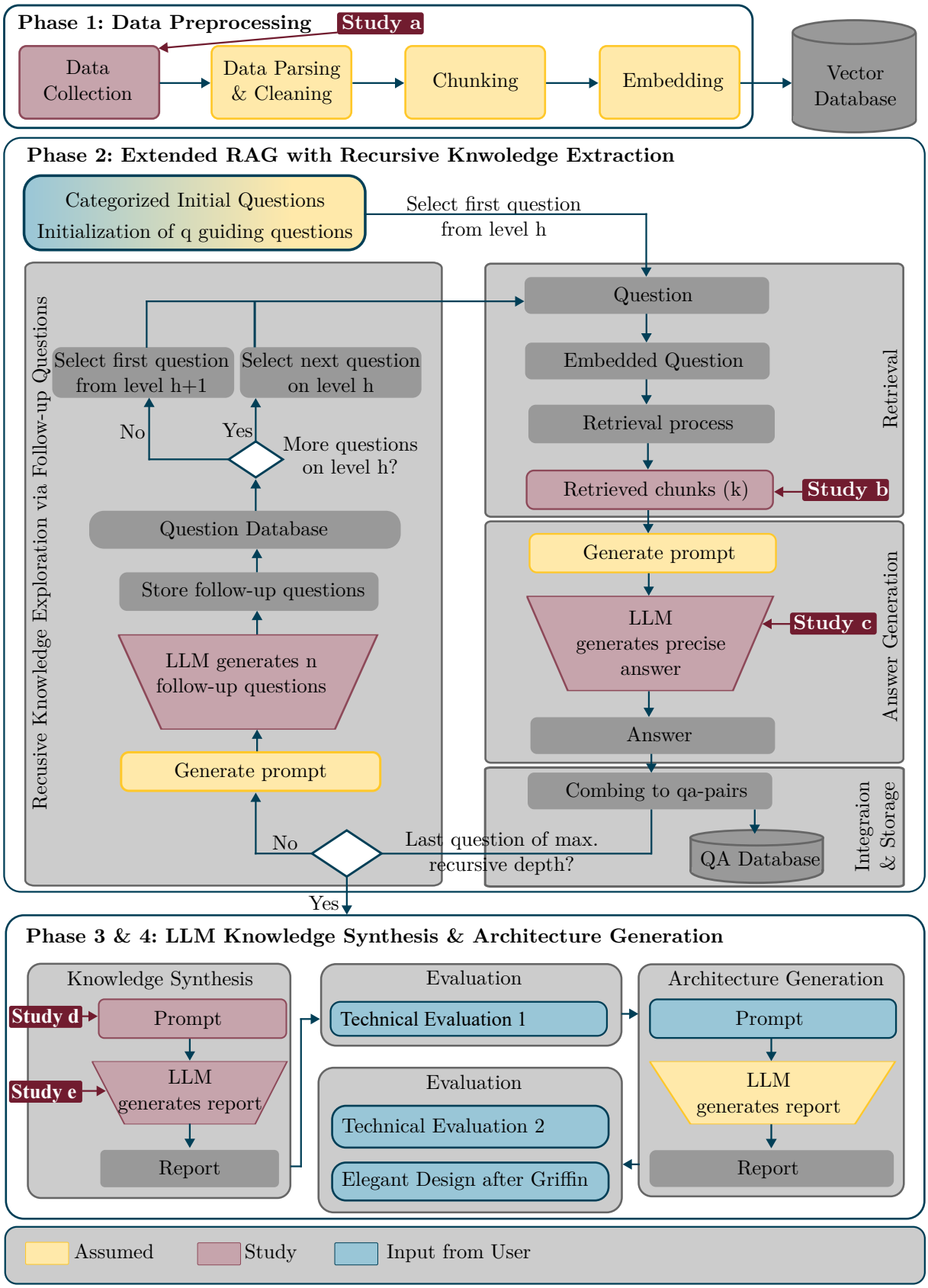

Figure 4.1.: Configurable parameters of the methodology across the four phases

A first central factor concerns the composition of the **Data Collection**, which defines the scope and type of documents. The corpus size and the mixture of scientific publications, patents, and technical documentation influence the breadth and quality of available knowledge. In practice, however, identifying and acquiring relevant literature presents a significant challenge, particularly in highly specialized technical domains. Access restrictions, confidentiality issues, and the fragmentation of knowledge across various databases often limit the availability of documents. The hybrid collection strategy described in Section 3.1.1 addresses these practical limitations while still enabling domain-specific adaptation of data selection (see Phase 1 in Figure 4.1), where this investigation is flagged as "Study A". The scope and particularly the relevance of the source corpus are directly related to the question of optimal vector store composition investigated later in Section 4.2.1. This study analyzes how the vector store size and composition influence the percentage of queries that can be successfully answered.

For **Data Parsing**, a hybrid strategy is implemented, based on the systematic evaluation in Section 3.1.3. The comparative analysis of various parsers, as documented in Table 3.2 on page 57, shows significant performance differences depending on the document type and content. For the technical systems addressed in this dissertation, LlamaParse with GPT-4.1 proved optimal for complex scientific documents with many equations, LlamaParse with GPT-4.1-mini for standard text extraction, and OCRmyPDF for scanned patent documents. It should be emphasized, however, that this selection is not universally transferable. For other domains or special document types, such as structured requirements tables, other parser models may be more advantageous. The choice of parser model should therefore always be evaluated individually based on the specific characteristics of the document, particularly since parser capabilities are constantly evolving.

The **Chunking** strategy represents another parameter. Various methodological approaches exist for document segmentation, from simple length-based methods (fixed token or character count) to paragraph-based segmentation to more complex semantic segmentation. In this dissertation, the semantic segmentation approach described in Section 3.1.4 was deliberately chosen, as it preserves contextual relationships and recognizes natural thematic transitions. This property is particularly valuable for complex technical documentation, where related concepts, system components, or process descriptions are often explained across multiple paragraphs. Although this implementation has proven effective in practice, no systematic study has been conducted to optimize the segmentation parameters.

For the **Embedding** configuration, the model "*sentence-transformers/all-MiniLM-L6-v2*" was employed, which serves in practice as a pragmatic compromise between semantic expressiveness and efficiency. This choice was primarily based on practical applicability and availability, not on a comprehensive evaluation of various embedding models, see Section 3.1.4. A more rigorous model selection could involve domain-specific benchmark tests, where different embedding models are compared using representative technical queries, and the retrieved results are rated by experts for relevance. Alternatively, retrieval quality could be assessed via a self-consistency approach that measures how reliably a model retrieves thematically related documents. These optimization steps were beyond the scope of this work but represent promising directions for future research.

The **Vector Database** includes several configuration aspects that have a significant influence on retrieval performance. ChromaDB was chosen as the vector database system for this dissertation, although other widely available alternatives, such as Pinecone, Qdrant, Weaviate, Milvius or Faiss, are also possible options [181]. ChromaDB was selected for several reasons. It offers a simple Python integration, supports both in-memory and persistent storage, and enables flexible extension with metadata filtering, a crucial advantage for the category-based query strategy employed in this work. A particularly critical parameter is the chosen *distance metric*, which determines how semantic similarity between vectors is calculated. The cosine similarity was implemented, which measures the

angle between vectors and thereby emphasizes directional proximity in semantic space. Alternative metrics include Euclidean distance and Manhattan distance.

4.1.2. Parameters of the Extended RAG-System

The retrieval process, which involves recursive question generation, forms the core of the developed RAG system and differs substantially from conventional single-query-based approaches, as shown in Section 3.2. The performance of this process is determined by several key parameters, which are explained in more detail below.

The **Categorized Initial Question Structure** serves as the starting point for systematic knowledge exploration. In this work, a total of six categories (RFLP*) have been used, covering the various aspects of technical systems. Five initial guiding questions were defined per category, serving as a starting point for recursive exploration. It is important to note that the categorization can be flexibly adapted to the specific requirements of different development projects. Depending on the development object, methodological strategy, and analysis focus, alternative or supplementary categorizations are conceivable. For more software-oriented systems, for example, additional categories such as data structures, interfaces, or security aspects could be integrated. The categorization chosen here thus represents a basic framework that can be adapted on a project-specific basis without compromising the methodological foundation of the recursive knowledge acquisition system. To operationalize this framework, each category was initialized with five guiding questions, providing a structured, yet cognitively manageable starting point for recursive exploration. This number was chosen based on cognitive considerations, as five questions fall within Miller's cognitive range while ensuring manageability for human-machine interaction.[1] The chosen number proved practicable in the use cases carried out, but has not been further evaluated systematically. Determining the ideal number of categories and question density represents an interesting topic for future research. In particular, it could be explored whether domain-specific category systems or adaptive question catalogs that dynamically adapt to the state of knowledge could lead to improved knowledge extraction. However, the exponential increase in questions with each recursion level (as shown in Figure 3.10) sets practical limits that must be considered when developing such optimization approaches.

The **Retrieval Process** is based on a vector-based semantic search and represents a particularly influential parameter, which is examined in detail in Study B (Section 4.2.2). In Phase 2 of Figure 4.1, this evaluation is indicated as "Study B" at the "retrieved chunks (k)" stage. The k-value, which determines the number of retrieved chunks, was systematically evaluated with values of 5, 10, 20, and 40 to determine the optimal compromise between information completeness and precision.

The **Prompt Generation** for interaction with the LLM is another essential parameter that significantly influences the quality of generated answers. The prompt structure used in this work was designed according to principles of prompt engineering and is described in detail in Section 3.2.3 (page 66). This includes the role of instruction as a domain-specific expert, the precise formulation of the request, the structured presentation of the context from the retrieved chunks, and specific instructions for answer generation, including an explicit request for source citation and chain-of-thought documentation. This prompt

[1] Miller's "magic number" of 7±2 items describes the human short-term memory capacity [182]. It has been widely applied across various domains like team sizes [183, p.71] and document structuring principles [184, pp.55-56]. Recent research questions the universal validity of this concept, emphasizing that cognitive capacity depends on task complexity as well as context and is often smaller than 7 [184, p.54]. Nevertheless, this established guideline provides a reasonable starting point, leading to the choice of five questions (7-2) for initial systematic exploration in this dissertation.

structure was standardized for the entire system and is based on established best practices in dealing with LLMs for information-extracting tasks. Although no dedicated parameter study was conducted for prompt optimization, the system's performance indicates that the chosen structure is effective in generating precise, source-based, and traceable answers. The standardization of the prompt across all queries also ensures consistency in system performance and comparability between different answers.

A further central configuration factor is the **LLM configuration**, which can be divided into two aspects: the choice of the model and the tuning of its generation parameters. The model selection is systematically investigated in Study C (Section 4.2.3), comparing several state-of-the-art LLMs[2] (GPT-4.1-mini, GPT-4o, Claude-3.7-sonnet, Claude-3.5 Haiku, Gemini-1.5-Flash, and Gemini-2.0-Flash) with respect to their ability to process technical content precisely. In Phase 2 of Figure 4.1, this comparison is indicated as "Study C" within the answer-generation lane. In addition to model choice, specific generation parameters also influence the quality and precision of the outputs. For example, the temperature setting also plays a role. For answer generation, a low temperature of 0.1 was implemented, based on the assumption that minimal randomness in token selection is advantageous for fact-based answers. This decision is based on the premise that in technical domains, the precise reproduction of information extracted from documents should take priority. A low temperature reduces the likelihood of hallucinations or creative deviations from the source material, leading to more predictable, factually grounded responses while preserving the inherently probabilistic nature of language generation. This configuration was combined with the chain-of-thought technique, which makes the model's thinking process explicit and thereby improves the traceability of answers. It should be noted that this temperature setting was not optimized through a dedicated empirical study but is based on theoretical considerations and documented best practices for fact-oriented RAG applications. The system's positive performance with this configuration indicates that the chosen value of 0.1 is adequate for the given application. However, systematic parameter optimization in future work could enable further improvements.

The **Prompt for Follow-up Question Generation** follows a specific format that is fully documented in Section 3.2.4 (page 66). The prompt instructs the LLM to act as a technical analyst, developing more profound follow-up questions based on the previous QA-pair. The structure includes precise role instructions for analyzing technical documentation, presenting the original QA-pair as context, specific instructions for generating a fixed number of follow-up questions, and guidelines for the thematic orientation of these questions. The prompt explicitly requests questions that target concrete design parameters, component relationships, design decisions, implementation details, and performance characteristics, thereby ensuring a systematic deepening of technical domain knowledge. This prompt structure was maintained for all recursion levels, ensuring consistent exploration quality across the entire knowledge space.

The **Recursive Follow-up Question Generation over the LLM** controls the exploratory aspect of the system. The parameter for the number of follow-up questions per answer determines the breadth of exploration at each recursion level. In contrast, the recursion depth limits the maximum depth of knowledge exploration. In this dissertation, the number of follow-up questions (n) was set to two per answered question, representing a compromise between the breadth of exploration and system complexity. A larger number of follow-up questions would allow for broader thematic coverage. Still, it would also lead to an exponential increase in the total number of questions, as illustrated in Figure 3.10 (page 67). For follow-up question generation, the same LLM model was used as for answer generation, but with a slightly increased temperature of 0.2. This moderate increase in temperature aims to enable more creative and explorative questioning without

[2]Selection reflects the state of the art between March and May 2025.

venturing too far into speculative territory. This choice is based on a theoretical consideration of balancing exploration and factual accuracy, but was not evaluated through a dedicated parameter study. Future investigations could systematically evaluate both different temperature settings and various values for the number of follow-up questions (n) to determine the optimal configuration for other domains and use cases.

4.1.3. Parameters of Knowledge Synthesis and Architecture Generation

Phases 3 and 4 of knowledge synthesis and architecture generation are primarily controlled by three adjustable main parameters: model selection, temperature setting of the LLM, and the used prompts.

For the **Knowledge Synthesis**, a low temperature (0.2) was used to achieve high factual accuracy when generating the synthesis report according to the RFLP* scheme and classifying information in the three-level confidence classification system (Green/Orange/Red). The structural elements themselves, the RFLP* scheme, and the classification system are implemented as system components of the prompt and were not parameterized. Two factors significantly influence the quality of the generated report within this phase: (i) prompt structure and length, and (ii) LLM selection. "Study D" (Section 4.2.4) isolates the impact of prompt length and modularization on structural clarity, requirements capture, and interface coverage. These are the most critical aspects for ensuring the quality of knowledge synthesis. "Study E" (Section 4.2.5) evaluates model performance under identical input and formatting conditions. The comparison considers differences in context-window capacity, latency, and cost, as these factors directly influence both feasibility and output quality. The remaining synthesis workflow components (RFLP* scheme, confidence coding, and HTML skeleton) serve as fixed system elements and therefore require no separate parametric studies.

For the **Architecture Generation** of new solutions, the GPT-o1 model was selected for its significantly reduced input token requirements compared to the knowledge synthesis phase and its chain-of-thought reasoning capabilities that are particularly valuable for complex architectural design tasks. Unlike other LLMs, the GPT-o1 model does not expose temperature parameters, as its internal reasoning mechanism automatically balances analytical rigor with thinking based on the specific requirements of each architectural challenge. This self-regulating approach eliminates the need for manual parameter calibration while ensuring that generated architectures maintain engineering feasibility. Boundary conditions and constraints are specified explicitly in the input prompt, not through adjustable system parameters.

The **Evaluation** of those two phases represents a methodological step whose scope and depth can vary depending on the use case, but does not contain explicitly configurable system parameters. Expert feedback is incorporated through manual adjustments to the knowledge synthesis report before architecture development. Overall, the configuration options in this phase are deliberately kept lean, as the focus is on methodological integration of automated knowledge synthesis and human expertise, rather than complex parameterization of the process.

The parameters summarized in Table 4.1 on the next page reflect the controllable configuration points across the four methodological phases. Each parameter directly influences retrieval effectiveness, synthesis quality, or architectural consistency.

Table 4.1.: Classification of system parameters and their methodological influence

Parameter	Description	Influence
Phase 1: Data Processing		
Data Collection	Defines corpus scope and composition	Establishes the knowledge foundation for all subsequent phases. Missing content cannot be recovered in later phases. Detailed analysis in Study A.
Data Parsing	Extracts text from different document formats	Affects structure preservation and readability. Parser choice must match document characteristics.
Chunking	Segments documents into coherent semantic units	Controls contextual granularity and preserves logical continuity.
Embedding	Converts text chunks into vector representations	Balances semantic expressiveness and efficiency. Defines similarity precision.
Vector Database	Stores and retrieves embeddings via similarity search	Shapes retrieval precision and filtering. The metric choice governs the structure of access to information.
Phase 2: Recursive Retrieval and Question Expansion		
Categorized Initial Questions	Structured domain exploration via RFLP* scheme	Ensures systematic coverage across different system aspects.
Retrieval Process	Number of retrieved chunks (k) per query	Balances completeness and retrieval noise. Defines factual basis of answers ($\rightarrow$ Study B).
Prompt Generat.	Specifies structure and role of LLM prompts	Controls consistency, traceability, and contextual precision of generated content.
LLM Selection	Determines reasoning capability and factual accuracy	Influences the depth and precision of answers ($\rightarrow$ Study C).
LLM Temperature	Controls randomness of factual responses	Low values reduce hallucinations. The temperature is set at 0.1.
Follow-up Generation	Number (n) and recursion depth for follow-ups	Governs exploration depth and scalability of recursive knowledge acquisition.
Phase 3: Knowledge Synthesis		
Prompt Length	Structural clarity and completeness during synthesis	Determines coherence, hierarchy, and the overall quality and consistency of the synthesized output ($\rightarrow$ Study D).
LLM Selection	Affects synthesis accuracy and efficiency	Governs contextual integration and semantic precision of synthesized knowledge ($\rightarrow$ Study E).
LLM Temperature	Balances creativity and factuality during synthesis	T = 0.2 ensures factual robustness with slight generative flexibility.
Evaluation Procedure	Output evaluation	Provides quality assurance and plausibility review of knowledge report.
Phase 4: Architecture Generation		
LLM Selection	Generates new system architectures from synthesized knowledge	Affects reasoning transparency and technical feasibility of architectures.
Evaluation Procedure	Output evaluation	Provides quality assurance and plausibility review of generated architectures.

4.2. Systematic Parameter Studies

While established configurations from the literature or practical experience exist for some of the mentioned parameters, particularly critical parameters such as the choice of LLM model or the prompt length require systematic optimization for the specific use case of generating technical documents. The parameters highlighted in red in Figure 4.1 have a particularly strong influence on the quality and efficiency of the overall system and were therefore specifically investigated.

Within the scope of this dissertation, five central studies on the most critical influencing factors were conducted, which are presented in detail in the following sections:

- Study A: Scaling effects of vector store size on the number of answerable questions
- Study B: Influence of the number of retrieved chunks (k) on the completeness and precision of information extraction ($k = 5$, 10, 20, 40)
- Study C: Comparison of various state-of-the-art LLM models regarding answer quality (GPT-4o, GPT-4.1-mini, Claude-3.7-Sonnet, Claude-3.5-Haiku, Gemini-1.5-Flash, Gemini-2.0-Flash)
- Study D: Influence of prompt length on answer quality
- Study E: Comparison of various LLM models for knowledge synthesis

These studies form the basis for developing a configuration strategy tailored to the specific requirements of technical system architecture generation. The methodological approach and the results of the individual studies are presented in detail below.

4.2.1. Study A: Vector Store Size and Number of Answered Queries

The scope and composition of the vector store represent a parameter that significantly influences the RAG performance. While a smaller, highly focused knowledge base may provide precise but limited answers, a larger, comprehensive collection could offer broader coverage, but at the potential cost of introducing noise and reducing retrieval precision. This study examines the impact of three different vector store configurations, with the hypothesis that a targeted, domain-specific strategy can achieve high performance by balancing knowledge depth with retrieval accuracy.

Experiment Implementation

To systematically investigate the effects of vector store size and composition on answer coverage and availability, a multi-stage experimental setup was developed to ensure comparability and reproducibility of the results. In an initial run, a standardized question corpus was generated. Starting from 30 high-level questions covering various aspects of technical systems (five questions for each of the six RFLP* categories), the recursive question process was executed over five hierarchy levels, with two sub-questions per answer. This process generated a total of 930 questions and answers. For the actual study, the same 930 questions were applied in all subsequent runs, but answered using different vector store configurations. This prerequisite ensured that observed differences could be attributed primarily to the variation in vector store size and scope, rather than to differences in the underlying question set. In all runs, other configuration parameters of the methodology were kept constant. In particular, the LLM model (GPT-4.1-mini), the temperature setting (0.1), the k-value (20), and the prompt structure remained unchanged.

Keeping these factors constant was essential to eliminate interfering influences and to isolate the effect of the vector store configuration.

This study utilized a thermal management system (TMS) dataset as a representative example of domain-specific technical literature. The three tested vector store configurations were:

1. Top 20 TMS Papers (Aircraft Focus): A curated selection of 20 aircraft-focused TMS papers, identified based on the multi-dimensional relevance assessment described in Section 3.1.1.

2. All Aircraft TMS Papers: A comprehensive collection of all available TMS papers related to aircraft applications, comprising 58 specialized documents covering various aspects of aircraft TMS.

3. Complete Literature: The entire compiled knowledge base, comprising 115 documents, including aircraft standards, TMS patents, TMS papers, and general TMS applications, representing the most comprehensive but potentially noisy dataset.

930 answers were generated for each of the three vector store configurations, totaling 2,790 answers.

Results and Analysis

The analysis of the results shows a clear relationship between the vector store configuration and the coverage of answerable questions (see Figure 4.2). It is important to note that any answer that did not contain the indication "*No answer possible based on the provided documents*" was counted as an "*answered question.*" This means that the LLM was able to generate an answer based on the provided documents. The percentage of answered questions is entirely dependent on the dataset used and will vary with other document collections. The following observations were made:

- Knowledge base size effect: The percentage of answered questions increased with the vector store size, from approximately 72% for the "Top 20 TMS Papers" to 85% for "All TMS Papers", and reaching 93% for the "Complete Literature" collection. These values represent the mean answer coverage across all six RFLP* categories. The percentages shown in Figure 4.2 indicate the category- and hierarchy-level-specific answer coverage, defined as the ratio of answerable questions to the total number of questions within the identical question set.

- Category-specific performance: Different categories showed varying sensitivity to vector store size. Categories like RQ (Requirements) and FU (Functions) benefited significantly from the expanded knowledge base, while categories like AP (Architecture Pattern) showed more modest improvements.

- Complexity dependence: Higher hierarchy levels (more detailed questions) showed greater improvement with larger vector stores, indicating that detailed queries benefit more from comprehensive context availability.

- Marginal improvement: The increase in coverage from "All TMS Papers" to "Complete Literature" was noticeably smaller than the improvement from "Top 20 TMS Papers" to "All Aircraft Papers". This pattern can be interpreted as an indication of a potential saturation effect. However, with only three tested dataset sizes, it should be considered as a descriptive observation of this dataset rather than evidence of a generalizable trend.

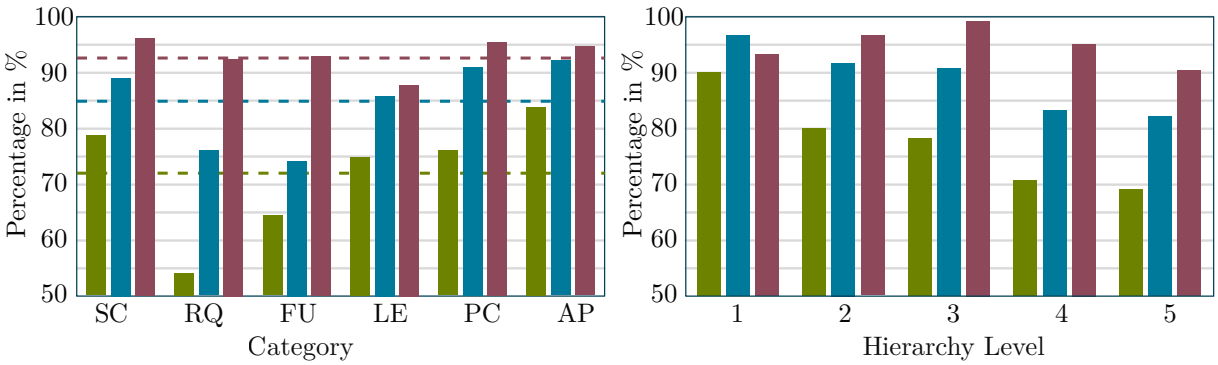

Figure 4.2.: Percentage of answered questions by category (left) and hierarchy level (right) for different vector store configurations. Average answer coverage: 72% (20 TMS Papers), 85% (All TMS Papers), 93% (Complete Literature). Categories: SC = System Context, RQ = Requirements, FU = Functions, LE = Logical Elements, PC = Physical Components, AP = Architecture Patterns.

Building on these marginal improvements, an additional observation concerns the scaling behavior of the uncovered share with respect to vector store size. The results suggest an approximate "halving pattern": each doubling of the dataset size reduces the proportion of unanswered questions by about 50%. Formally, this can be expressed as

$$U \approx U_0 \cdot \left(\tfrac{1}{2}\right)^d, \tag{4.1}$$

where U represents the remaining share of unanswered questions after d successive dataset doublings. U_0 is the initial proportion of unanswered questions for the smallest dataset configuration. In the present study, the uncovered share decreased from 28% (20 documents, 72% answered) to 15% (58 documents, 85% answered) and further to 7% (115 documents, 93% answered). These values are in high agreement with the expected results of 14% and 7% derived from the halving model. This indicates that the reduction of unanswered questions resembles an exponential decay, with diminishing absolute improvements at larger dataset sizes, and suggests that the system approaches a saturation region where additional documents yield only marginal gains. A possible explanation could be that the still unanswered questions tend to concern more specific pieces of information that occur only in a smaller portion of the literature. If the coverage of such rarer topics similarly increases with corpus size, this could account for the observed halving pattern. However, given only three data points, this should be regarded as an initial observation whose underlying cause would need to be examined more closely in future work with a more fine-grained dataset.

Conclusions for System Configuration

Based on the results of this study, the following configuration recommendations can be derived for domain-specific RAG applications:

- For applications with strict computational constraints, a highly curated selection of the most relevant documents provides acceptable coverage (72%).
- For balanced performance with domain-specific focus, a comprehensive collection of all available domain-relevant papers represents an optimal compromise, achieving

high coverage (85%) while maintaining topical relevance and manageable computational overhead.

- For maximum answer completeness, the complete literature configuration, including standards, patents, and broader applications, is recommended, achieving near-complete coverage (93%) at the cost of potential retrieval noise.

It is essential to note that the optimal vector store configuration depends significantly on the specific use case and the distribution of questions. Furthermore, it should be noted that the compiled literature does not represent an optimally curated dataset but consists exclusively of publicly available sources, reflecting a deliberately more challenging and realistic testing condition rather than an idealized best case. The recommendations presented here are based on experimental testing with technical domain-focused questions and are therefore most relevant for similar knowledge-intensive, domain-specific applications. Based on these results, the complete literature configuration was used to generate all subsequent reports and for further investigations in this work.

4.2.2. Study B: Analysis of the k-Value

The number of most relevant document chunks (k) returned from the vector store for a given query, and provided as context for the LLM, represents a critical parameter. A value that is too small may result in missing important information, whereas a k-value that is too large may introduce noise and impair the model's accuracy. The study investigates four different k-values (5, 10, 20, 40) with the hypothesis that a medium value might be optimal as it offers a balance between information completeness and precision.

Experiment Setup

To systematically investigate the effects of the k-value on the number of answerable questions, a multi-stage experimental setup was developed to ensure the comparability and reproducibility of the results. In an initial run, a standardized question corpus was generated. Starting from 30 high-level questions covering various aspects of technical systems (five questions for each of the six categories). The recursive question process was executed over five hierarchy levels, with two sub-questions per answer. This process generated a total of 930 questions and answers. For the actual parameter study, the same 930 questions were used in all subsequent runs, but answered with different k-values (5, 10, 20, and 40). As in Study A, using the identical question set was essential to maintain the validity of the study and to avoid systematic biases. Generating different follow-up questions introduces uncertainty in each run, making it difficult to isolate the influence of the k-value. By using an identical question set, it is ensured that the observed differences in answer quality are solely attributable to the variation of the investigated parameter (k) and not to differences in the questions asked. In all runs, the other system parameters were kept constant. The used LLM model (GPT-4.1-mini), the temperature setting (0.1), the prompt structure, and the vector store configuration. This is essential to eliminate confounding variables and reliably measure the isolated effect of the k-value. 930 answers were generated for each of the four k-values, totaling 3,720 answers.

Results and Analysis

The analysis of the results shows a clear relationship between the k-value and the number of answerable questions, see Figure 4.3. It is important to note that any answer that

did not contain the indication "*No answer possible based on the provided documents*" was counted as an "*answered question*." This indicates that the LLM was capable of generating a substantive answer based on the provided documents. The percentage of answered questions depends entirely on the dataset used and will vary across document collections. The reported percentages represent the mean value across all categories (SC to AP) for each k-value. In particular, the following observations were made:

- Completeness: With increasing k, the number of answers marked as "answered" improved continuously:
  - 73.7 % at $k = 5$
  - 86.5 % at $k = 10$
  - 93.2 % at $k = 20$
  - 95.9 % at $k = 40$
- Category-specific differences: The evaluation reveals clear variations across categories. At $k = 5$, PC performs best (79.4%), whereas RQ achieves the lowest value (61.3%). For $k = 20$, all categories achieve values above 87%.
- Complexity dependence: For broader questions (hierarchy level 1), there was a smaller difference between the k-values than for more profound questions at higher hierarchy levels, indicating a stronger context dependence of deeper queries.
- Saturation effect: The percentage improvement decreases with increasing k-value. The most significant jump occurs between $k = 5$ and $k = 10$ (+12.8 percentage points), while the difference between $k = 20$ and $k = 40$ is only 2.7 percentage points.

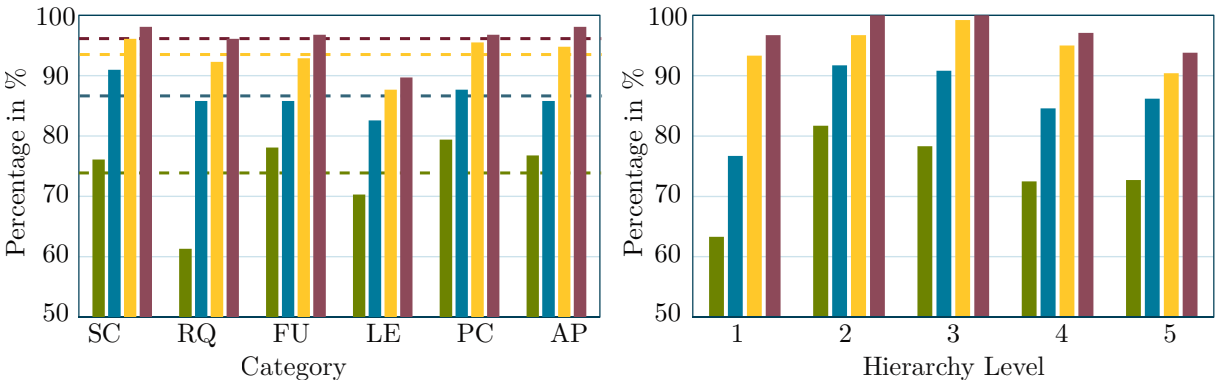

Figure 4.3.: Percentage of answered questions by category (left) and hierarchy level (right) for different k-values. Average answer coverage: 73.7% at $k = 5$, 86.5% at $k = 10$, 93.2% at $k = 20$, 95.9% at $k = 40$.

Quantitatively, the trend exhibits a similar halving behavior to that described in Equation 4.1, although the scaling variable differs. While Equation 4.1 expresses the reduction of the uncovered share as a function of dataset size, the present results reveal an analogous relationship with respect to the k-value. The unanswered questions decrease from 26.3% at $k = 5$ (as the complement to 73.7% answered at $k = 5$) to 13.5% at $k = 10$, 6.8% at $k = 20$, and 4.1% at $k = 40$, closely matching the expected halving pattern for each doubling of k. Only at the highest k-values, a slight saturation effect becomes visible, indicating an asymptotic lower bound of the remaining unanswered fraction.

Conclusions for System Configuration

Based on the results of this study, the following configuration recommendations were derived:

- For answers with acceptable quality, $k = 10$ is a sensible setting, as 86.5% of answers were already rated as complete.
- For use cases with high quality requirements, $k = 20$ is recommended, as 93.2% achieves very high completeness without requiring the increased computational effort of $k = 40$.

It is noteworthy that in this dataset, k=20 yielded the most balanced trade-off. However, this observation cannot be interpreted as a generalizable optimum. In fact, it remains an open question whether a universal k-value exists at all, or whether the optimal setting will always depend on the specific use case, dataset characteristics, and model configuration. The recommendations presented here are therefore limited to the tested scenario with aircraft-specific technical documents and should be primarily viewed as guidance for comparable applications. Based on these results, $k = 20$ was consistently used for generating all subsequent reports and for further investigations in this work, as this value represents the best compromise between answer quality and computational effort for the given document collection.

4.2.3. Study C: Comparison of Different LLM Models for Answer Generation

The selection of the LLM for answer generation is also a critical parameter within the RAG pipeline, significantly influencing the quality of the generated reports. In this study, six leading LLMs are compared in terms of their ability to create precise, factually accurate, and technically correct answers based on retrieved document fragments.

For the systematic evaluation, a specialized framework based on LangSmith[3] was used. This framework ensures reproducibility through standardized test conditions, enables direct comparability of models under identical conditions, offers complete transparency through logging of all process steps, and supports automated evaluation through specialized evaluators. In line with the evaluation scheme introduced in Section 2.4.3, the semantic criteria are operationalized via an LLM-as-a-Judge method [138, 139]. Specifically, LangSmith's prebuilt evaluator prompts follow a standardized format and can be found in Appendix A.6 on page 192. The evaluators use a uniform evaluation scale of numbers between one and five, with higher values representing better results. They are implemented as exportable components, enabling reuse for future model evaluations. Four central criteria were used to evaluate the answers:

- Factuality: Evaluates the correct reproduction of facts and technical details from the source documents. The rating scale ranges from 1 (significant factual errors) to 5 (complete factual correctness).
- Completeness: Measures how comprehensively relevant information from the source documents is considered in the answer. The scale ranges from 1 (missing essential information) to 5 (complete coverage of all pertinent information).

[3]LangSmith is a platform for observing, evaluating, and monitoring AI applications. The platform enables developers to debug and test AI applications and transfer them from the prototype to the production phase. https://smith.langchain.com/

- Source Fidelity: Evaluates the correspondence with the provided sources and the avoidance of extrapolations or hallucinations. The scale ranges from 1 (significant non-source-based claims) to 5 (perfect source fidelity without unsubstantiated statements).
- Technical Precision: Evaluates the correct use of domain-specific terminology and technical concepts. The scale ranges from 1 (incorrect technical terminology) to 5 (precise and consistent technical language).

The test dataset used for evaluation comprises a total of 12 questions, each with the corresponding retrieved chunks, evenly distributed across the six identified categories (two questions per category).

Experiment Implementation

The experiment was designed as a controlled comparative study with identical input conditions. Six LLMs were included in the investigation: GPT-4o (multimodal flagship model), GPT-4.1-mini (more efficient GPT-4 variant), Claude-3.7-Sonnet (current Anthropic flagship model), Claude-3.5-Haiku (speed-optimized variant), Gemini-1.5-Flash, and Gemini-2.0-Flash (both optimized for speed and efficiency).

To ensure maximum comparability, several conditions were kept constant. All models answered the same 12 questions, received identical document chunks as context for each question, were configured with a consistent temperature setting of 0.1, and received the same task prompt, see the prompt in Section 3.2.3.

The evaluation process followed a standardized procedure with four consecutive phases. In the first phase, all models received the identical task prompt. The second phase involved the actual answer generation, with all models operating at a constant temperature setting of 0.1. In the third phase, each generated answer was systematically evaluated by the implemented evaluators along the four defined quality dimensions. The concluding fourth phase involved the comparative analysis of the results to identify significant performance differences between the models and to derive model-specific characteristics.

Box 6: Illustrative Example of Model Evaluation (Excerpt)

```
Input (excerpt):
Question: "What are the system boundaries of aircraft ECS and which aircraft
           systems  directly interface with them?"
Context: 10 retrieved document chunks like "Chunk 1: Additionally, it interfaces
         with the aircraft's water separation and drainage systems...", Chunk 2:
         "Integration with the aircraft's pressurization system is critical for
         maintaining proper cabin conditions throughout the flight ...", etc.

Claude-3.7-Sonnet - Output (excerpt):
Answer: "Based on the provided documents, the aircraft ECS interfaces directly
         with: 1. The aircraft's water separation and drainage systems [Ref:
         1aa4cce8-35fe-46a2-bf65-96efbef05b50] 2. The aircraft's pressurization
         system [Ref: 1aa4cce8-35fe-46a2-bf65-96efbef05b50] 3. The pressurized
         zones of the fuselage (cockpit, cabin, and forward cargo) through the
         supply network [Ref: da129c93-ed94-455b-98f7-4a06550bac51]. ECS may
         also interface with power sources which can be either pneumatic or elect.
         [Ref: 1aa4cce8-35fe-46a2-bf65-96efbef05b50], though the documents don't
         provide a comprehensive list of all system boundaries and interfaces."
```

The experiment resulted in a total of 72 generated answers (12 questions × 6 models), which were systematically evaluated using LangSmith. To illustrate the evaluation process, Box 6 provides an excerpt of one representative test case. Each model received the same input (consisting of a standardized question and ten retrieved document chunks as contextual evidence) and generated an answer based solely on this information. The responses were subsequently assessed using the four defined evaluation criteria (factuality, completeness, source fidelity, and technical precision).

Results and Analysis

The evaluation was conducted based on the four defined parameters, supplemented by performance and cost metrics. Cost calculations are based on the respective providers' token-based pricing models, which account for both input tokens (prompt and context) and output tokens (generated responses). It should be noted that pricing is subject to frequent updates by the providers. Therefore, the reported costs reflect the conditions on the specific day of measurement, when all experiments were conducted in a single session to ensure consistency. Latency is measured as the end-to-end response time from the initiation of the API request to the complete reception of the response, encompassing model inference time, API processing overhead, and network transmission delays. All reported values in Table 4.2 represent mean scores (arithmetic averages) calculated across the 12 evaluated questions per model. The quantitative results are summarized in Table 4.2, based on the precise measurements from LangSmith.

Table 4.2.: Performance metrics of various LLMs in the RAG system

Model	**Factuality**	**Tech. Prec.**	**Source Fidelity**	**Completeness**	**Latency (s)**	**Costs ($)**
GPT-4.1-mini	5.00	5.00	5.00	5.00	22.52	0.02
GPT-4o	5.00	4.92	5.00	4.83	13.44	0.12
Claude-3.7-Sonnet	5.00	5.00	5.00	4.92	17.19	0.21
Claude-3.5-Haiku	5.00	5.00	5.00	4.83	12.98	0.05
Gemini-1.5-Flash	5.00	4.83	4.92	4.92	5.57	0.01
Gemini-2.0-Flash	4.83	4.67	4.83	4.00	3.21	0.01

The results reveal a high performance level for all tested models, particularly in the dimension of factuality, where five of the six models achieve the highest value of 5.0. Nevertheless, characteristic differences emerge in the various quality dimensions as well as in latency and costs. GPT-4.1-mini is the only model that achieves the maximum value of 5.0 in all four dimensions. This result demonstrates the high suitability of the model for precisely processing technical content in the RAG context. However, the comparatively high latency of 22.52 seconds, i.e., the time between request and complete answer generation, is notable as the highest among all tested models. This response time is particularly relevant for system architecture applications where iterative queries might be posed. The costs are nevertheless very moderate at 0.02$.

GPT-4o achieves maximum values for factuality and source fidelity as well as very high values for technical precision (4.92) and completeness (4.83). The latency is significantly lower than that of GPT-4.1-mini at 13.44 seconds, but at six times higher costs (0.12$).

Claude-3.7-Sonnet shows excellent overall performance with nearly perfect values. With maximum factuality, technical precision, and source fidelity, as well as very high completeness (4.92), it offers a balanced performance profile for technical system architecture

questions. The latency is in the middle range at 17.19 seconds, but the costs are the highest in the test at 0.21$.

Claude-3.5-Haiku offers impressive performance despite its optimization for efficiency, with maximum values for factuality, technical precision, and source fidelity. For completeness (4.83), there is a slight lag. With a latency of 12.98 seconds and moderate costs of 0.05, it offers an attractive balance between performance and efficiency.

The Gemini models show contrasting characteristics: Gemini-1.5-Flash delivers excellent values for factuality (5.0) and completeness (4.92) with slightly reduced values for technical precision and source fidelity, while Gemini-2.0-Flash shows somewhat lower values in all dimensions, particularly in completeness (4.0). The strength of both models lies in their efficiency with extremely low latencies (5.57 and 3.21 seconds, respectively) and minimal costs (0.01 each$).

Table 4.2 presents the quality metrics of all models in comparison, along with their associated costs. Particularly striking is the discrepancy between performance and costs: While the qualitative differences between the top models are relatively small, the costs vary by more than twentyfold. This insight is particularly relevant for use cases with high throughput or limited budget. The latency values show apparent differences between the tested models: the Gemini Flash models offer the lowest latencies (3-6 seconds), Claude-3.5-Haiku and GPT-4o fall in the middle range (13-14 seconds), while GPT-4.1-mini and Claude-3.7-Sonnet exhibit the highest values (17-23 seconds).

Conclusions for System Configuration

The study shows consistently high performance across all tested models, with main differences in latency and cost. Within the evaluated dataset (March-May 2025), GPT-4.1-mini achieved the maximum score of 5.0 in all four quality dimensions, demonstrating strong suitability for the intended use case, particularly in terms of factuality, source fidelity, and completeness. However, these results reflect performance at the time of study and should not be interpreted as universally valid, since model capabilities evolve rapidly and outcomes may vary with different datasets or evaluation settings.

Despite its high quality, GPT-4.1-mini remains cost-efficient (0.02$ per run), representing only a fraction of the most expensive model (Claude-3.7-Sonnet, 0.21$) and offering an excellent quality-cost ratio. As a newer model variant, GPT-4.1-mini also benefits from the latest advances in AI architecture, promising better long-term support and compatibility with future system extensions, thus ensuring future viability.

While other models may offer advantages for specific application scenarios, such as Gemini-2.0-Flash for latency-critical applications or Gemini-1.5-Flash for cost-sensitive environments with high throughput, GPT-4.1-mini provides the best overall performance for the primary use case of generating precise and reliable technical answers.

All tested models reached consistently high quality scores in this evaluation, demonstrating their suitability for technically demanding RAG applications. However, the significant differences in latency and cost among models with comparable quality highlight an increasing differentiation in the market, where efficiency factors complement quality considerations. Future evaluations should therefore be extended to a larger and more diverse question set as well as additional engineering domains to validate these findings and assess the generalizability of model performance across different technical contexts.

4.2.4. Study D: Influence of Prompt Length on Report Quality

The study compared modular and monolithic prompt structures to assess their effectiveness in generating a knowledge synthesis and an architecture generation report. The central question is whether sequential processing of individual RFLP* aspects in separate, focused prompts leads to qualitatively better results than a comprehensive monolithic prompt that addresses all aspects simultaneously.

Experiment Setup

Two distinct prompting strategies were implemented and compared. The *monolithic* strategy employed a single, comprehensive prompt that combined the instructions of both modular prompts into one. It simultaneously contained all guidance for generating the system context and the corresponding requirements, requiring the LLM to process both aspects within a single execution. In contrast, the *modular* strategy utilized the same two prompt parts, but executed them sequentially as separate, specialized prompts. The first prompt focused exclusively on the system context, while the second prompt concentrated on requirements generation with reference to the previously generated document.

To ensure a valid comparison, all remaining parameters were held constant. Both experiments received identical input data in the form of the same QA-pairs from knowledge extraction, used the same LLM model (GPT-4.1), employed the same temperature setting (0.3), and required the same output format specifications (HTML with color coding).

The modular prompt used for requirements generation is shown in Box 7, illustrating the focused, aspect-specific instruction set that characterizes the modular approach. The corresponding system context prompt is provided in Appendix A.7 on page 195.

Evaluation Criteria

In the absence of established metrics for LLM-generated technical documents, this study specified five criteria covering both quantitative and qualitative aspects, all assessed manually. Structural quality refers to the clarity, consistency, and logical organization of the output. Content completeness captures the coverage of relevant aspects (number of requirements) and the depth of detail. Technical precision reflects the correctness and specificity of technical specifications and parameters. Traceability concerns the transparency of derivations and source referencing. Finally, coherence between partial aspects denotes the logical connection between Requirements and System Context.

Results and Analysis

The main difference between the approaches concerns the amount and granularity of the generated content. The modular approach generated 55 detailed requirements compared to only 18 requirements from the monolithic approach, representing a threefold increase in content volume. The quantitative results are summarized in Table 4.3, which shows consistent advantages of the modular approach across all categories except functional requirements, where both approaches are equal. A requirement was counted if it appeared as a distinct bullet/entry in the document. Obvious duplicates or paraphrases were consolidated manually to avoid double-counting. Both sets of requirements were technically plausible and logically consistent within their respective report contexts, indicating that the difference between the approaches primarily concerns the level of detail and completeness rather than the conceptual validity of the generated content.

Box 7: Modular Prompt Example (Excerpt)

```
Create a comprehensive report of the REQUIREMENTS of an Aircraft Environmental
Control Systems (ECS) based on the provided documents. The report should include:

Requirements of the Conventional ECS
- Provide a comprehensive narrative description of all ECS requirements in flowing
  paragraph form, covering all performance requirements, safety requirements,
  reliability requirements, environmental requirements, certification
  requirements, interface requirements, and user requirements.
- EXCEPTION TO GENERAL FORMAT: After completing the narrative paragraphs, add a
  section titled "Summary of Key ECS Requirements:" that presents all requirements
  as a bullet-point list.
- Each requirement should be listed as a separate bullet point with concise wording
- Ensure all quantitative values and specifications from the paragraphs are
  included in this summary list.

## Format and Presentation:
* Create a detailed, technically precise HTML document
* Use tables, diagrams, lists, and other structuring elements
* Use the following color coding for transparency:
  - <span style="background-color: #90EE90;">GREEN</span>: Information directly
    from the provided documents [with specific source reference and qa-pair ID]
  - <span style="background-color: #FFD700;">ORANGE</span>: Information derived
    or combined from the documents [with justification and source qa-pair IDs]
  - <span style="background-color: #FFB6C1;">RED</span>: Newly developed or assumed
    information [with detailed explanation of assumpt. and engineering reasoning]
* Include a table of contents with hyperlinks to sections
* Ensure the report is exceptionally thorough and technically precise
* For each requirement, provide:
  - Unique identifier
  - Requirement statement
  - Quantitative parameters (where applicable)
  - Source/origin
  - Priority/criticality level

## CRITICAL INSTRUCTIONS:
* When information is missing or incomplete, make reasonable assumptions based on
  aircraft engineering principles, but highlight all such assumptions
  with ORANGE or RED background color.
* Use the provided documents from DATABASE as the primary source of information
  and ensure that all relevant information from these documents is integrated.
* etc.
```

Tables 4.4 and 4.5 present excerpts of the LLM-generated HTML documents that show the external interface specification generated by the monolithic approach (Table 4.4) and the modular (Table 4.5), respectively. These excerpts are illustrative samples. Based on them, three observations are noteworthy:

- Coverage: In the excerpt, the Table 4.4 generated with the monolithic prompt lists four interfaces, whereas the table that was generated with the modular prompt shows six (see Table 4.5).
- Source attribution: The table that was generated with the modular prompt (4.5) provides a source column with QA-pair IDs (one-to-one traceability). The monolithic table gives aggregated sources below the table (e.g., [SC.2, SC.5.1, SC.4.2, SC.2.2.2]), which do not map to specific interfaces.

- Quantitative parameters: Both tables include ranges (e.g., pressure, temperature, flow) and implement the required color-coding system for source transparency. However, the modular approach was applied more consistently and systematically, enhancing the traceability of information sources and derivation paths.

Table 4.3.: Comparison of requirements coverage by category

Category	**Modular**	**Monolithic**	**Difference**
Functional	4 Req	4 Req.	0
Performance	20 Req.	5 Req.	+15
Safety	10 Req.	3 Req.	+7
Reliability	5 Req.	2 Req.	+3
Environmental	5 Req.	3 Req.	+2
Interface	6 Req.	1 Req.	+5
User	5 Req.	2 Req.	+3
Total	**55 Req.**	**18 Req.**	**+37**

Table 4.4.: Excerpt of the HTML document that was generated with the monolithic prompt, showing the external interfaces of the ECS

External Interface	**Description**	**Quantitative Parameters**	**Protocols/Standards**
Engine/APU Bleed Air	Hot, high-pressure air supply	30-45 psig; 177-232°C (max 260°C); flow ∼0.55-0.65 lb/min/passenger	Physical pneumatic; control via discrete signals (assumed)
Ram Air System	Ambient air for cooling	-55°C to +50°C; pressure varies with altitude/speed	Physical ducting, valve actuation (assumed)
Electrical System	Power for ECS controls, valves, fans	28V DC/115V AC	ARINC 429/629 or CAN bus (assumed)
Cabin Pressuriz.	Outflow valve, pressure sensors	Cabin diff. pressure: 8-9 psi typical	Discrete/analog signals (assumed)

Table 4.5.: Excerpt of the HTML document that was generated with the modular prompt, showing the external interfaces of the ECS

Interface	**Type**	**Parameters**	**Protocol/Format**	**Source**
Engine/APU Bleed Air	Physical (pneumatic)	Temp: 180-250°C, Pressure: 30-45 psig, Flow: ∼0.6 lb/min/passenger	Mechanical ducting, analog/digital sensors (protocol not specified)	[SC.1] [SC.2], [SC.5.1]
Ram Air	Physical (ambient)	Temp: -55°C to +50°C, Flow: variable	Mechanical ducting, flow/temperat. sensors	[SC.2], [SC.4.2]
Electrical System	Electrical	Voltage: 115V AC/28V DC, Power: per ECS component	Standard A/C power bus, digital control signals	[SC.5]
Cabin Pressuriz.	Physical/ control	Cabin pressure set-point, out-flow valve position	Analog/digital control signals	[SC.1]
Water Drainage	Physical	Condensate flow, drain line heating	Mechanical piping, electr. heating control	[SC.1], [SC.5]
ACMS/Health Monitoring	Data	Temp., pressure, humidity, airflow, valve positions, turbine speed	Data bus (e.g., ARINC 429/664, not specified), digital data	[SC.3.1]

Beyond the quantitative differences, the qualitative analysis revealed significant disparities in technical precision and structural organization. Although both strategies received the same information basis through identical QA-pairs, the modular approach consistently produced more precise technical specifications with specific, measurable parameters. For instance, the modular output specified "*ECS shall maintain turbine pressure ratio between 2.5:1 and 4:1, and secondary compressor pressure ratio between 1.3:1 and 1.8:1*," while the monolithic output provided statements such as "*ECS shall maintain appropriate pressure ratios in turbine and compressor components*." The modular version delivers specific, testable parameters suitable for actual system design, whereas the monolithic version remains too general for practical implementation.

Regarding structural organization, the modular approach produced consistently well-organized documents with clear hierarchical structures and uniform formatting. Each requirement followed a standardized template including unique identifiers, verification methods, and source references. The monolithic approach, while covering all essential topics, exhibited inconsistent depth across sections and less systematic organization. The findings reflect a single dataset, a specific model configuration (GPT-4.1, temperature 0.3), and a predefined prompt design. They should be interpreted as descriptive evidence for this setup rather than a universal rule. Different domains, models, or context budgets may lead to different outcomes.

Interpretation and Theoretical Cause Analysis

The analysis reveals distinct qualitative differences between the two prompting strategies. The observed quality differences can be attributed to several theoretical and methodological factors:

- Attention distribution: For complex tasks, simultaneous processing of multiple aspects within a single prompt leads to a dilution of the model's self-attention. The limited computational budget must be distributed across all tokens in the context window, which fragments attention and reduces the achievable depth of detail. In contrast, the modular approach allows the full attention capacity to be concentrated on one RFLP* aspect at a time, resulting in denser and more complete elaborations.
- Task complexity management: Decomposing the task into smaller, well-defined modules reduces overall complexity per generation step. Each sub-prompt can contain highly specific and optimized instructions tailored to its aspect, while a monolithic prompt necessarily remains more generic and therefore less directive.
- Trade-off between coherence and depth: Both strategies yield structured outputs, but with different emphases: the modular approach achieves greater structural consistency and traceability, whereas the monolithic approach preserves stronger immediate coherence between partial aspects due to single-pass generation.

The results align with principles of the Cognitive Load Theory (CLT), which was originally developed by Chandler and Sweller [185] in the 1980s for human learning processes. CLT postulates that the working memory has limited capacity and that optimal performance occurs when this capacity is not overwhelmed by excessive cognitive demands [185, p.293]. When the total cognitive load exceeds the available processing capacity, performance degrades significantly, leading to superficial processing and reduced output quality. Although LLMs do not learn in the human sense, they exhibit analogous capacity constraints in the form of fixed context windows and token budgets. The modular prompting strategy effectively applies the CLT principle of load management through task segmentation: by decomposing a complex reasoning task into smaller, sequentially processed modules, it minimizes context interference and enables deeper, more focused analysis within each step.

Conclusions and Recommendations

In this evaluation, modular prompts yielded higher structural clarity, more complete requirement capture (55 vs. 18), and more consistent traceability than a monolithic prompt. The systematic separation of RFLP* aspects into sequentially processed modules results in a significant quality improvement. For the practical implementation in this dissertation, the following recommendations are therefore derived:

- Complex knowledge synthesis and system architecture tasks should be divided into clearly defined, focused modules that are processed sequentially.
- Each module should reference and integrate the results of preceding modules.
- The prompt structure should be specifically optimized for each module to address the peculiarities of the respective RFLP* aspect optimally.

This modular prompt strategy has proven to be more robust and qualitatively higher for both knowledge synthesis and system architecture generation, forming the methodological basis for all further developments in this work.

4.2.5. Study E: Evaluating LLM Models for Knowledge Synthesis

Within the methodology, a systematic evaluation of various LLM models regarding their suitability for generating a technical document was conducted. The study focused on the models' ability to generate a structured system context document that meets the requirements of the RFLP* framework.

Investigated LLMs

As part of the analysis, a total of eight different LLM models were investigated. Three models, GPT-4.1-mini, Gemini-2-0-Flash, and GPT-4.1, successfully processed the knowledge synthesis and were comprehensively evaluated accordingly. The remaining five models (Claude-3-7-Sonnet, Claude-Haiku, GPT-o3-Mini, GPT-o4-Mini, GPT-o1-Mini) proved unsuitable for the task due to token limitations. The prompt used with extensive context information from 930 QA-pairs exceeded the maximum input context window of these models. The complete prompt totaled 204,163 tokens, which significantly exceeded the available context windows ranging from 128K to 200K tokens for the excluded models, as shown in Table 4.6. The decisive factor here was not the number of QA-pairs alone, but the combined length of each question together with its corresponding answer, which determined the overall token count. Since token usage increases linearly with input length, the prompt exceeded the maximum context window of these models and thereby disqualified them from processing such extensive technical documentation tasks.

Table 4.6.: Comparison of analyzed LLM models regarding their context window. Data from [186].

	GPT-					**Gemini**	**Claude**	
Token Limit [Tokens]	**4.1-mini**	**4.1**	**o3-mini**	**o4-mini**	**o1-mini**	**2.0-Flash**	**3.7-Sonnet**	**Haiku**
Input context window	1M	1M	200K	200K	128K	1M	200K	200K
Max. output length	32K	32K	100K	100K	65.5K	8K	128K	8K

Evaluation Criteria

For the systematic evaluation of the generated document, several qualitative and quantitative criteria were used. The analysis began by examining the structure and organization of the output HTML documents, with a particular focus on semantic correctness, navigation ergonomics, and hierarchical consistency. Simultaneously, the content depth and thematic coverage of the documents were evaluated to ensure that all required topic areas were addressed in accordance with RFLP* process. Technical accuracy and consistency formed another key criterion, examining the precision of technical parameters, unit consistency, and the correct representation of system relationships. Furthermore, the proper use of color coding and source referencing was analyzed, which is essential for the transparency and traceability of a technical document. The color coding scheme, as shown in Table 4.7, distinguishes between information directly retrieved from the source documents (green), derived or inferred based on context (orange), and hypothetical or assumed content (red). Visual presentation and readability were also included in the evaluation. Finally, objective performance metrics such as processing time and cost per query were determined to quantify the efficiency of the various models.

Results and Analysis

To illustrate the qualitative differences, Table 4.7 shows the excerpt generated by GPT-4.1, Figure 4.4 presents the corresponding section from GPT-4.1-mini, and Table 4.8 depicts the output produced by Gemini-2.0-Flash. Each excerpt covers the same ECS interface section and was generated using identical input data and prompt structure, enabling direct comparison of formatting, structural fidelity, and technical precision. Across the three models, clear qualitative and structural differences became apparent. All implementations correctly applied the predefined color-coding scheme (green for directly retrieved, orange for derived, and red for assumed information), yet the degree of adherence to the requested table structure and precision of technical content varied considerably.

GPT-4.1 and Gemini-2.0-Flash both produced the requested tabular format with high structural coherence, whereas GPT-4.1-mini generated the section as an unstructured list, which reduced readability and hindered direct comparison across parameters. Among the structured variants, GPT-4.1 produced the most consistent and traceable document. Its table featured a uniform column layout, explicit per-row source references, and clearly

Table 4.7.: Excerpt of synthesized HTML table generated by GPT-4.1

Interface	Parameters	Protocols/Formats	Direction	Source
Engine/APU Bleed Air	Pressure (30-45 psig), Temp. (180-250°C), Flow Rate (1.8 kg/s)	Pneumatic, mechanical (no digital protocol)	Engine/APU → ECS	[SC.1, SC.5.1]
Ram Air System	Ambient pressure/ temperature, flow rate (modulated by valves)	Physical airflow (controlled by ECS actuators)	External → ECS	[SC.1, SC.5.1]
Electrical System	Voltage, power supply (aircraft bus)	Electrical wiring, standard aircraft power bus	Aircraft → ECS	[SC.1, SC.5.1]
Cabin Pressuriz. System	Cabin pressure set-points, valve positions	Signal lines (analog/digital); Likely ARINC 429 or simil.	ECS ↔ Pressuriz.Sys.	[SC.1]
ACMS/Maintenance	Sensor data (temp., pressure, humidity)	Data bus (protocol not spec., likely ARINC 429/ CAN)	ECS ↔ ACMS	[SC.3.1, SC.3.2]
Structural Interfaces	Physical mounting, duct sealing	Mechanical	ECS ↔ Airframe	[SC.1]

The ECS interfaces with multiple aircraft systems, including:

- Engine Bleed Air System: Primary energy source for air cycle ECS, supplying compressed hot air.
- Ram Air System: Provides cooling air for heat exchangers.
- Electrical System: Powers control valves, sensors, actuators, and fans.
- Cabin Pressurization System: Controls cabin pressure via outflow valves.
- Water Separation and Drainage Systems: Manage condensate removal.
- Structural Pressure Vessel: ECS ducts pass through bulkheads and structural interfaces.

While specific interface protocols and detailed data exchange formats are not explicitly provided, the ECS control unit centrally manages these interfaces through sensor inputs and control signals to maintain system stability and performance [Ref: SC.1, SC.5, LE.5].

Figure 4.4.: Excerpt of synthesized HTML report generated by GPT-4.1-mini

defined parameter ranges with appropriate units. Gemini-2.0-Flash also generated a coherent HTML table with inline references and detailed quantitative information, though minor irregularities in formatting and color application slightly affected overall uniformity. GPT-4.1-mini covered all required interfaces conceptually but lacked consistent column alignment and cross-referencing, resulting in a lower level of organization.

Table 4.8.: Excerpt of synthesized HTML table generated by Gemini-2.0-Flash

Interface	Parameter	Typical Value/Range	Format
Engine Bleed Air	Pressure	30-45 psig [Ref: SC.5, SC.5.1]	Pneumatic
Engine Bleed Air	Temperature	200-230 °C (after pre-cooler) [Ref: SC.5.1], up to 260 °C (max) [Ref: SC.2]	Pneumatic
Engine Bleed Air	Flow Rate	0.55-0.65 lb/min per pax, 1.8 kg/s total [Ref: SC.5.1]	Pneumatic
APU Bleed Air	Pressure	Assumed similar to bleed air: 30-45 psig	Pneumatic
APU Bleed Air	Temperature	Assumed similar to engine bleed air: 200-230 °C (after pre-cooler)	Pneumatic
Ram Air System	Temperature	-55 °C to +50 °C [Derived from SC.2, SC.4.2]	Pneumatic
Ram Air System	Pressure	Varies with altitude and flight speed [Derived from SC.4.2]	Pneumatic
Electrical System	Voltage	Assumed 28 V DC or 115 V AC 400 Hz	Electrical
Electrical System	Power	Variable (dep. on valves, fans, compressors)	Electrical
Cabin Pressuriz.	Cabin Pressure	Equivalent to 6,000-8,000 ft altitude. Derived from RQ.4	Pneumatic
ACMS	Data Parameters	Temperature, pressure, humidity, air-flow, valve positions, turbine speeds [Derived from SC.3.1]	Communic. protocols not specified, assumed ARINC 429 or similar
Flight Crew Interface	Temperature Setpoints	Assumed range: 18-24°C	Analog or digital, specific protocol not specified

With regard to technical precision, GPT-4.1 achieved the most balanced results. It specified testable parameter ranges and maintained internal consistency across quantitative values. Gemini-2.0-Flash provided extensive numerical data, especially for interface parameters, but sometimes conflated derived and assumed information, indicated by a higher share of orange- and red-coded entries. GPT-4.1-mini produced technically plausible but less detailed statements, reflecting the inherent limitations of its list-based format.

Performance measurements further supported these qualitative observations. GPT-4.1 completed the generation fastest (1.24 s) at moderate cost ($0.36 per run). Gemini-2.0-Flash was the most cost-efficient ($0.04 per run) but exhibited slightly higher latency. GPT-4.1-mini was markedly slower (113.35 s) and the most expensive ($0.70), making it less suitable for iterative or large-scale documentation. Despite these variations, all three models generated technically coherent and contextually valid documents under the given experimental conditions. The observed differences, therefore, concern structure, traceability, and presentation quality rather than factual correctness.

It should be noted that the synthesized reports represent deterministic snapshots of model behavior at the time of execution. Although the same prompt and configuration were used in each run, minor variations in wording or structure may still occur due to the inherent stochasticity of LLM text generation. Such differences remain within an acceptable tolerance and do not affect the overall qualitative assessment presented in this study.

Discussion and Conclusions

The comparative evaluation highlights clear performance and quality distinctions between the analyzed models. Among them, GPT-4.1 demonstrated the most balanced overall performance, combining structural precision, technical accuracy, and efficient processing. The generated document exhibited a coherent hierarchy, explicit source attribution, and consistent quantitative parameters, which are essential for technical traceability within the RFLP* framework. Gemini-2.0-Flash also achieved high-quality results and provided a cost-effective alternative, though its output showed minor inconsistencies in formatting and a slightly less rigorous differentiation between derived and assumed data. In contrast, GPT-4.1-mini produced a usable but less refined document: its list-based structure lacked formal organization, and its significantly higher cost and latency further limited its practical applicability for repeated or large-scale document tasks.

The exclusion of five additional models due to context-window limitations emphasizes a fundamental technical constraint for knowledge synthesis in engineering domains. The complete prompt used in this study, encompassing 930 QA-pairs and more than 200,000 tokens, exceeded the maximum input capacity of those models. This finding underlines that sufficient context length is a decisive prerequisite for domain-specific document tasks, where large volumes of cross-referenced information must be processed within a single reasoning cycle. While smaller-window models may still perform adequately when supported by retrieval-augmented or chunk-based approaches, such hybrid methods were intentionally excluded from this direct model comparison to preserve experimental clarity.

Overall, the study confirms a clear quality-efficiency trade-off. GPT-4.1 offers the highest structural and technical quality at moderate computational cost, Gemini-2.0-Flash minimizes expenses while maintaining acceptable accuracy, and GPT-4.1-mini reaches lower practical efficiency despite higher costs. Based on these results, GPT-4.1 is recommended as the preferred model for generating RFLP*-aligned technical documents, particularly where precision, transparency, and structural consistency are critical. Gemini-2.0-Flash can serve as a viable option for less complex or budget-sensitive tasks, while GPT-4.1-mini should primarily be reserved for exploratory or small-scale applications.

It must be noted that these findings represent a temporal snapshot of the model landscape as of May 2025. Given the rapid development of large language models and the frequent extension of their context windows, the relative performance of the analyzed systems is subject to change. Continuous re-evaluation at intervals of six to twelve months is therefore advisable to maintain an optimal balance between quality, efficiency, and cost in future engineering documentation workflows.

4.3. Limitations and Future Optimization Potential

The presented parameter studies offer valuable insights into optimizing the developed system but exhibit methodological limitations that could be addressed in future work. First, the parameters were investigated sequentially rather than as a multivariate optimization problem, meaning potential interaction effects between parameters were not systematically captured. Second, the optimization focused on individual parameters identified as critical, while other configuration aspects were pragmatically set. Third, the evaluation was limited to a specific domain and a limited corpus, which could restrict the generalizability of the results. For future research, several promising directions present themselves: (1) Conducting multivariate optimization studies that consider parameter interactions, (2) systematically investigating previously pragmatically configured parameters such as embedding models or chunking strategies, (3) developing adaptive parameterization approaches that automatically adapt to different domains or document types, and (4) extending the evaluation to diverse technical domains to validate generalizability.

4.4. Summary

This chapter addressed RQ3 through systematic empirical analysis of critical parameters within the SAGE methodology, establishing evidence-based guidelines for deploying the approach effectively in aircraft engineering applications.

Five systematic parameter studies examined configurations across all operational phases. Study A (Section 4.2.1) demonstrated that comprehensive literature corpora (115 documents) achieve superior question coverage (93%) compared to focused selections (72%), though quality-focused curation remains more important than quantity. Study B established k=20 as the optimal retrieval parameter, providing 93.2% question coverage while maintaining computational efficiency. Study C revealed that GPT-4.1-mini achieves high performance across all quality dimensions (5/5 rating for factuality, technical precision, source fidelity, and completeness) while maintaining cost-effectiveness, making it the optimal model choice for technical RAG applications. Study D provided insights into prompt engineering, demonstrating that modular prompt structures significantly outperform monolithic approaches for complex knowledge synthesis tasks. This finding suggests attention dilution effects in LLMs and offers practical guidance for structuring complex technical prompts. Study E confirmed model selection criteria for knowledge synthesis, establishing GPT-4.1 for comprehensive document processing within token limitations. The selected parameters can be seen in Figure 4.5 in relation to their respective phases.

The empirical analysis enables a data-driven configuration of the methodology instead of relying on intuitive defaults. The identified optimal parameters (k=20, GPT-4.1-mini, modular prompts, semantic chunking) form the basis for the subsequent evaluations in Chapters 5 and 6, ensuring that evaluation builds on empirically optimized system performance. These results deepen the understanding of methodology behavior in technical domains and offer practical implementation guidelines transferable to other engineering applications beyond aircraft systems.

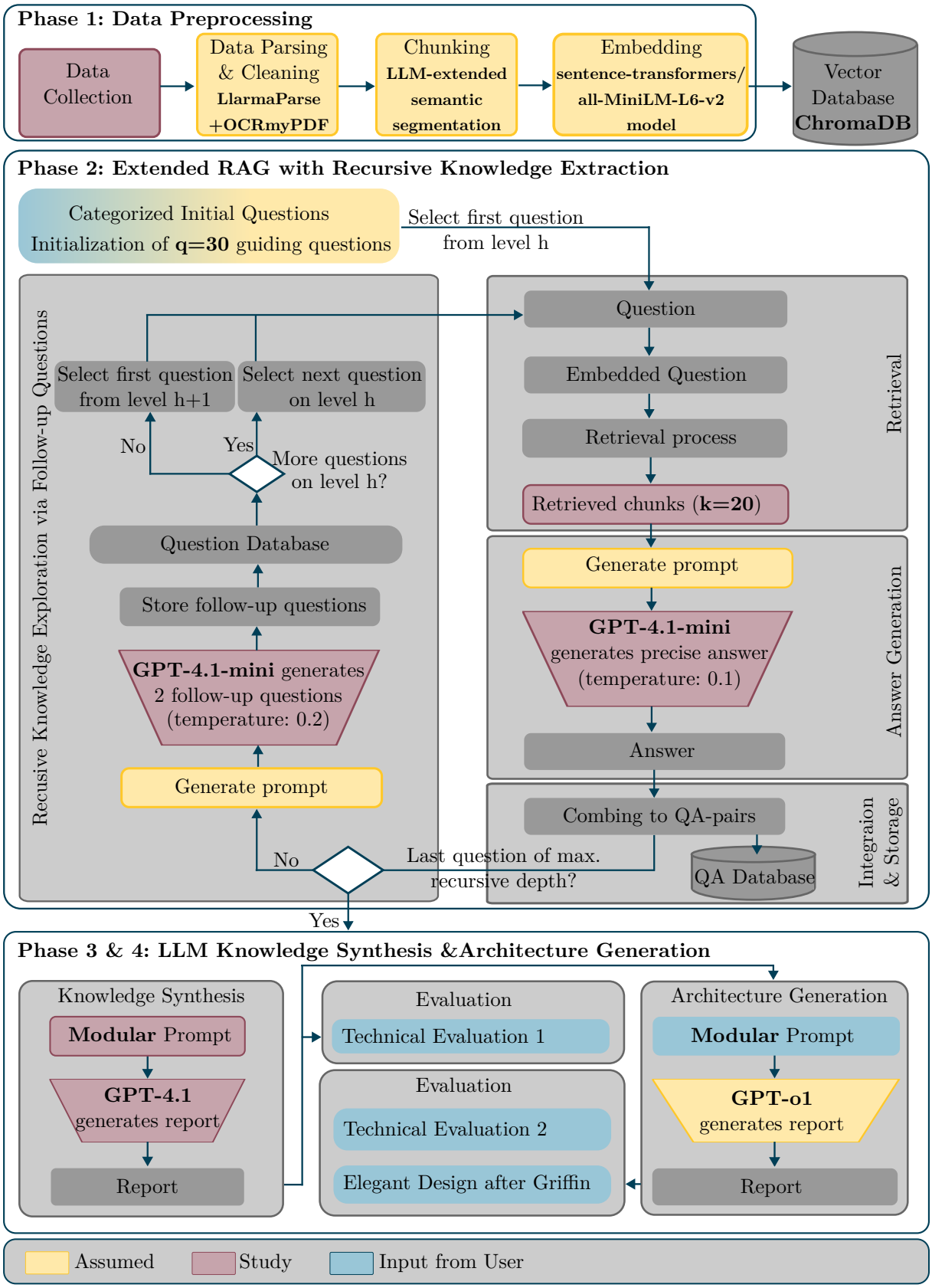

Figure 4.5.: Final system implementation showing the selected parameter values for the four main operational phases

5. Case Study 1: Environmental Control System

Addressing RQ1, RQ2, and RQ4: The evaluation of the methodology proceeds in three steps. RQ1, RQ2, and RQ4 introduced in Chapter 3 proposed methodological procedures for recursive retrieval and structured synthesis but have not yet been empirically tested. This case study provides the first opportunity to evaluate whether these mechanisms work in practice: (i) whether recursive retrieval can construct a comprehensive and traceable QA corpus from distributed literature (RQ1), (ii) whether LLM-based synthesis can transform this corpus into a coherent RFLP-structured document with accurate confidence classification (RQ2), and (iii) whether the resulting knowledge base can first support systematic documentation and component assessment through a Reuse/Adapt/New analysis and subsequently enable the generation of new, technically coherent system architectures (RQ4).*

As a prerequisite, this case study focuses on Environmental Control Systems (ECS). The following section provides a brief overview of both conventional bleed-air ECS and the electrically powered ECS (eECS) of the Boeing 787, which will later serve as an external reference system.

5.1. Use Case: Environmental Control System

The ECS of modern commercial aircraft satisfies complicated thermodynamic requirements under extreme environmental conditions. For instance, the cruising altitude of commercial aircraft is between 35,000 and 40,000 feet at air temperatures of -56°C and an air pressure of less than 0.25 bar [187]. Under these conditions, the ECS maintains a physiologically tolerable cabin pressure corresponding to a maximum altitude of 8,000 feet (equivalent to 1,800 - 2,400 m altitude), as specified in certification regulation CS-25.841 [87]. Another central requirement is temperature control. The system must be able to maintain the cabin temperature within a range of 18 to 30°C, taking into account different temperature zones. It compensates for the metabolic heat produced by passengers ($\approx$ 100 W per person) as well as the waste heat generated by onboard systems. Additionally, the ECS must ensure adequate air supply with a minimum fresh air provision of 0.25 kg/min per person (CS-25.831) while maintaining a complete air exchange every 2-3 minutes to guarantee cabin ventilation and acceptable air quality [87].

5.1.1. Conventional Bleed-Air ECS

The conventional ECS installed in most of the commercial aircraft is designed to regulate cabin pressure, temperature, humidity, and air quality. The system operates pneumatically, extracting air from the engines. This air is known as *bleed-air*. The functionality of the conventional ECS can be divided into the following main components and processes. The bleed air is extracted at high temperature (up to 425°C) and pressure (up to 25 bar)

from either the intermediate or high-pressure stage of the engine compressor [188, p.198], depending on the operating phase. After extraction, the air is directed to a primary heat exchanger, which reduces the temperature to approximately 180°C, using air from a ram air channel as the cooling medium. Subsequently, the air enters the air conditioning pack (see Figure 5.1), which, in its most common configuration, consists of two heat exchangers, a compressor, a turbine, and a water separator[1] [188, p.198]. Subsequently, the air passes through the primary heat exchanger and is then fed to the compressor, where it is compressed. The compressed, hot air is then directed to a secondary heat exchanger to lower its temperature further. Before the air enters the turbine, the condensed water from the cooled air is removed via a water separator. The air inside the turbine expands, leading to a significant drop in temperature. Afterwards, the conditioned air is directed to the mixing unit. If the air temperature is too low, trim air is used to increase the temperature, see Figure 5.1 [188], [190].

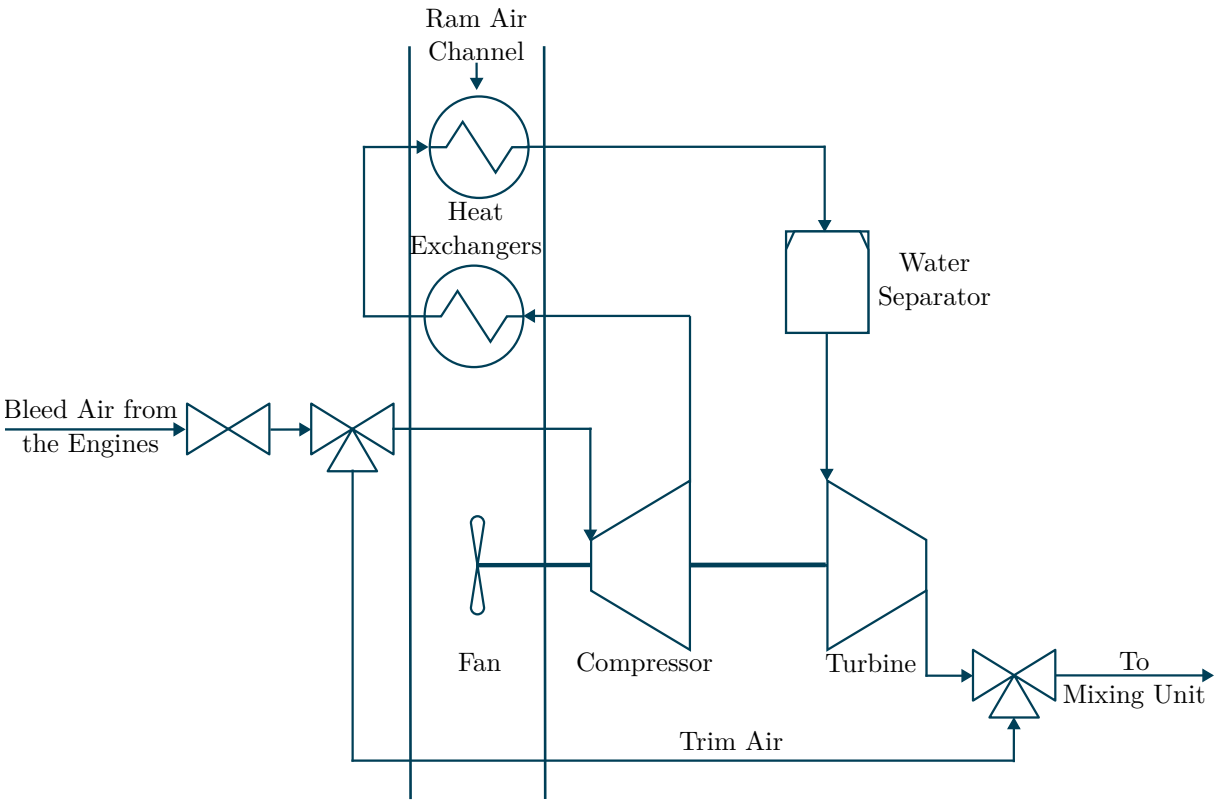

Figure 5.1.: Schematic structure of the ECS system in the A320

The fresh air is then mixed with recirculated cabin air in the mixing unit (typically at a 50:50 ratio) and directed into the cabin. The advantages of this system lie in its proven technology and in the use of compressed air generated by the engine. However, bleed air extraction leads to a considerable power loss in the engine and reduces its thermodynamic efficiency [191, p.23]. In addition, the system requires elaborate maintenance of the bleed air lines and carries potential risks due to the use of hot engine air [187].

These limitations of conventional bleed-air ECS have motivated research and industrial development toward alternative architectures that decouple the environmental control

[1]There are three main configurations of aircraft air conditioning packs. The simple air cycle (the most basic design), the bootstrap cycle (which has an additional compressor to increase efficiency), and the three-wheel bootstrap cycle (which has a fan, compressor, and turbine on a single shaft). The latter of which is the most common configuration found in modern commercial aircraft [189, pp.741-742].

function from the engines. The alternative to the bleed-air ECS is the electrically powered ECS of the Boeing 787, which serves in this study exclusively as an evaluation reference and was not included in the knowledge base.

5.1.2. Bleed-free ECS (eECS)

The Boeing 787 eECS fundamentally differs from conventional systems. Unlike the conventional system that draws bleed air from the engines, the B787 relies on an electric compressor, which offers several advantages in terms of energy efficiency and engine performance. The following description of the process flow in the eECS is based on the detailed system analysis provided by Yang et al. [192, pp.1373-1376]. Cabin air is drawn in through a ram-air duct located at the aircraft's fuselage. This drawn-in air, also known as *ram-air*, enters the electrically operated compressor, where it undergoes initial compression, increasing both temperature and pressure, as shown in Figure 5.2. Subsequently, the compressed air flows to the primary heat exchanger, where it is cooled under nearly isobaric conditions. After this initial cooling phase, the air enters the second compressor, where it undergoes further compression. This compressor is supported by the two downstream turbines T1 and T2, which are coaxially connected to it. The highly compressed air flows to the secondary heat exchanger for further temperature reduction. The airflow is subsequently routed through the regenerator and the condenser, resulting in the removal of moisture by condensation. In the subsequent water separator, the condensate is collected and injected into the cold side of the second heat exchanger to improve heat transfer. The dry air then flows through the cold side of the regenerator and is partially expanded in the T1 turbine, thereby cooling it. After passing through the cold side of the condenser, further expansion occurs in the second turbine. The conditioned air with a pressure above 0.9 bar and a temperature below 5°C is supplied to the mixing unit [192, p.1376]. Approximately 50% of the heated cabin air is recirculated and mixed with fresh, conditioned air. The remaining 50% is discharged into the atmosphere via the outflow

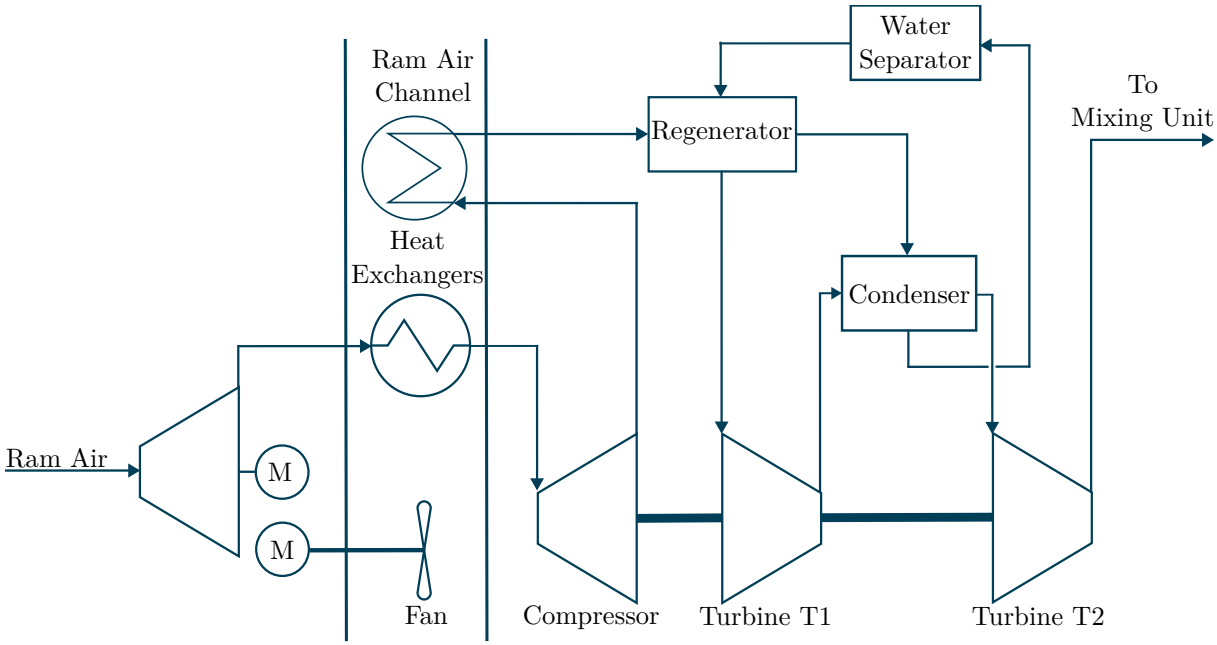

Figure 5.2.: Schematic structure of the eECS system in the B787, adapted from [192, p.1374]

valve. This continuous cycle ensures a constant cabin pressure altitude of up to 2,438 m (8,000 ft) at a pressure of at least 0.76 bar, as well as a comfortable temperature and air quality for passengers [192, pp.1373-1376]. This eECS serves as an evaluation reference while being deliberately excluded from the knowledge base.

Building on this technical foundation, the evaluation is conducted as a controlled experiment, where the development of an eECS is based solely on knowledge from conventional pneumatic ECS systems in the vector database. This approach offers three advantages. First, the B787 provides a real and well-documented reference system that allows an objective comparison between the generated and the actual design. Second, the evaluation examines whether the methodology can identify the limitations of conventional ECS architectures and propose technically consistent alternatives. Third, by deliberately excluding Boeing's eECS concept from the knowledge base, the experiment ensures that any generated design solutions originate solely from inferred reasoning rather than prior exposure. However, it cannot be ruled out that the underlying LLM encountered content related to the B787 during its general training. To minimize this risk, words like "Boeing," "B787," or "eECS" were never used in the prompts, to make no reference to the B787 at any point. This is the strictest isolation achievable within the given constraints.

In summary, this evaluation chapter follows a stepwise progression. Phase 1 preprocesses the selected ECS literature into a structured, searchable corpus (Section 5.2). Phase 2 applies recursive knowledge extraction to generate a comprehensive QA database (Section 5.3). Phase 3 synthesizes the fragmented QA-pairs into a coherent, RFLP*-structured document and evaluates this generated document (Section 5.4). Phase 4 builds on this document to conduct a Reuse/Adapt/New analysis and to develop alternative ECS architectures (Section 5.5). Together, these phases address whether systematic knowledge extraction and synthesis can provide a reliable engineering document (RQ1 and RQ2) and whether the methodology can generate technically coherent system architectures (RQ4).

5.2. Phase 1: Data Preprocessing

To enable systematic knowledge extraction, a dedicated literature corpus on conventional ECS was compiled and prepared for vector-based retrieval. Careful manual selection and filtering ensure that only information on bleed air systems is included, creating a clean starting point for evaluation.

5.2.1. Scope of Processed Data

For the knowledge extraction phase, a curated corpus of scientific literature on conventional aircraft ECS was compiled through thorough manual research. The data selection strategy prioritized depth over breadth, focusing on high-quality sources that provide comprehensive coverage of pneumatic system principles, components, and operational characteristics. The search strategy employed a hierarchical keyword matrix, combining primary keywords such as "aircraft environmental control system", "ECS aviation", "cabin pressurization system", "bleed air system", and "aircraft air-conditioning system" with secondary keywords such as "heat exchanger", "air conditioning pack", "cabin temperature control", and "pneumatic system".

For the present evaluation, a targeted selection was made from the identified literature corpus to fill the vector database with information on conventional bleed-air systems. The most critical aspect of this selection process was the rigorous exclusion of any literature containing information about eECS. This methodological limitation is essential for the

intended evaluation, as it ensures that the database contains no direct information about the concept being developed.

1. Selection and Exclusion Criteria of the Literature
The central challenge in creating the corpus was the need to identify and exclude documents with information on eECS and bleed-free systems. For this purpose, a filtering process was implemented:

1. Comprehensive manual keyword search: All documents were meticulously checked for terms such as “bleed-free”, “no-bleed”, “electric ECS”, “eECS”, “787”, and related terminology.

2. Detailed content review: Documents underwent thorough examination to ensure that no information, references, or comparisons to bleed-free systems were included, with particular attention to borderline cases.

3. Document modification: For documents that primarily addressed conventional systems but mentioned bleed-free systems only in isolated paragraphs or in the outlook section, these passages were completely removed. This ensured preservation of the relevant information on conventional systems without introducing eECS-related content into the corpus.

2. Quantitative Analysis of the Literature Corpus
After this evaluation and filtering process, the following view of the documents included in the analysis emerges: A total of 11 selected PDF documents were used as sources. These documents comprise 141 pages with an average of 12.8 pages per document.

The temporal distribution of the selected publications shows a clear focus on recent research. The collection spans from 2007 to 2024, with two publications from the 2007-2015 period representing 18.2% of the total. The majority of the selected documents are concentrated in more recent years, with five publications from 2016-2020 making up 45.5% of the collection, and four publications from 2021-2024 representing 36.4% of the total. This distribution reflects the emphasis on incorporating contemporary research findings and methodologies into the system.

3. Chunking and Vectorization
The data underwent a multi-stage preparation process that included text recognition, semantic segmentation, and vector transformation, as described in Section 3.1. After processing, a total of 985 optimized text chunks with extensive metadata were generated, which could be efficiently retrieved through the ChromaDB-based vector store. An example of one chunk can be seen in Box 8 on the next page.

5.3. Phase 2: Extended RAG

The following section details the configuration of Phase 2, using the enhanced RAG process. This process serves as the foundation for systematic knowledge acquisition from conventional ECS literature, creating the comprehensive question-answer database that will subsequently be synthesized into a coherent document in Phase 3.

Box 8: Example of a Chunk in the ECS Dataset (excerpt from Wright et al. [193])


```
{  "id": "01_2018_Wright_Aircraft_air_conditioning_merged_text_1",
    "text": "Keywords: Aircraft, Heat exchangers, Fouling, Lubrication oil
    contamination, Environmental control system. This paper describes how the
    aircraft's air-conditioning system functions, such as those on the A320
    including the components within the system. The need for an air-
    conditioning system requirement is explained, including the use of
    external airflow from outside to act as a cooling source (known in
    technical aviation terms as ram air). The A320 electronic controls are
    included in addition to the flight deck selections under normal
    operations and when the needs for higher levels of ventilation are
    required. The published literature identified indicates that the fouling
    and environmental pollution can affect the aircraft Environmental Control
    Systems is very limited, as there are few published or known materials
    about the particulate matter fouling/deposited on aircraft systems.
    Fouling collected from large commercial aircraft is analysed using
    different techniques to identify the potential source emission and
    composition. This paper addresses the failures of the ECS due to fouling,
    identifies the potential sources of fouling and operational measures that
    may affect this safety critical systems operation.",
    "metadata": {
      "document_id": "01_2018_Wright_Aircraft_air_conditioning_merged",
      etc. } }
```


5.3.1. High-Level Questions for Knowledge Synthesis

To gain a comprehensive understanding of existing ECS, a structured catalog of questions was developed. This catalog serves as the foundation for systematic knowledge acquisition and forms the starting point for the recursive question generation process.

The questions are divided into six categories, each addressing different aspects of the system. For clarity and brevity, only a small selection of representative questions is presented here, see Box 9. The complete question catalog with all guiding questions can be found in Appendix A.8 on page 196. The categories are the following:

1. System Context (SC): This category focuses on system boundaries, interfaces, and the operational environment of the ECS, addressing physical boundaries, operational parameters, stakeholder interactions, regulatory frameworks, and energy sources.

2. Requirements (RQ): This category addresses quantifiable specifications and constraints. The questions pertain to quantitative performance requirements for cabin temperature control, reliability requirements, general requirements for the ECS, safety requirements, and verification and evaluation methods for ECS certification.

3. Functions (FU): This category focuses on the capabilities and operational behavior of the system. The questions address the core functions of the ECS, measurement and monitoring functions, transition functions between different operational modes, safety and protection functions, as well as diagnostic and self-monitoring functions.

4. Logical Elements (LE): In this category, the focus is on system decomposition and information flows, examining functional subdivision into subsystems, airflow, thermal energy, and control signal paths, control architectures, information flow, and critical dependencies between ECS components and other aircraft systems.

5. Physical Components (PC): This category concentrates on hardware implementation. The questions address the main physical components of the ECS, specific air cycle machine designs, materials used, valve technologies, and the use of commercial-off-the-shelf components versus custom components.

6. Architecture Patterns (AP): This category deals with design patterns and system structures. The questions relate to reference architectures and design patterns, modularity, redundancy strategies, interface standardization, and architectural trade-offs that impact ECS performance.

The initial guiding questions serve as the starting point for the recursive knowledge acquisition process. They are submitted to the RAG pipeline, which extracts relevant information from the available literature and generates structured responses. Based on these responses, further, more specific follow-up questions are automatically generated to explore deeper aspects of the system.

Box 9: Representative Examples From the Category Requirements

```
System Context (SC):
SC.2: What are the specific operational envelope parameters for ECS design,
      including precise altitude ranges (in feet), external temperature limits
      (in Celsius), and maximum/minimum pressure differentials (in psi)
      across all flight regimes?
SC.5: What energy sources, bleed air, refrigerants, and other consumables are
      required for aircraft Environmental Control System operation

Requirements (RQ):
RQ.1: What are the quantitative performance requirements for cabin temperature
      control, including setpoint accuracy (Celsius), zone-to-zone temperature
      variation limits, and responsiveness metrics (time to achieve setpoint)
      across  different flight phases?
RQ.4: What specific safety requirements and thresholds govern the ECS design?

Functions (FU):
FU.1: What are the core functions of aircraft Environmental Control Systems?
FU.3: What functions allow the ECS to transition between ground operations,
      climb, cruise, descent, and emergency modes?

Logical Elements (LE):
LE.1: How are aircraft Environmental Control Systems decomposed into functional
      subsystems and what are their primary responsibilities?
LE.3: What control architectures (centralized, distributed, redundant) are used
      in aircraft Environmental Control Systems, and how is control logic implem.?

Physical Elements/Components (PC):
PC.1: What are the essential physical components of aircraft Environmental
      Control Systems and their typical specifications?
PC.3: What materials are used for critical Environmental Control System
      components in aircraft, and what properties make them suitable for
      aviation applications?

Architectural Patterns (AP):
AP.1: What reference architectures and design patterns are commonly used for
      Environmental Control Systems in different aircraft types?
AP.3: What redundancy strategies are implemented to ensure reliability in
      aircraft Environmental Control Systems?
```

5.3.2. Statistical Analysis of Question-Answer Generation

Following the recursive knowledge retrieval process described in Section 3.2, a comprehensive set of QA-pairs was automatically generated using the developed methodology. The statistical breakdown of the QA dataset provides insights into the coverage and distribution of answerable questions across the knowledge dataset of 11 documents.

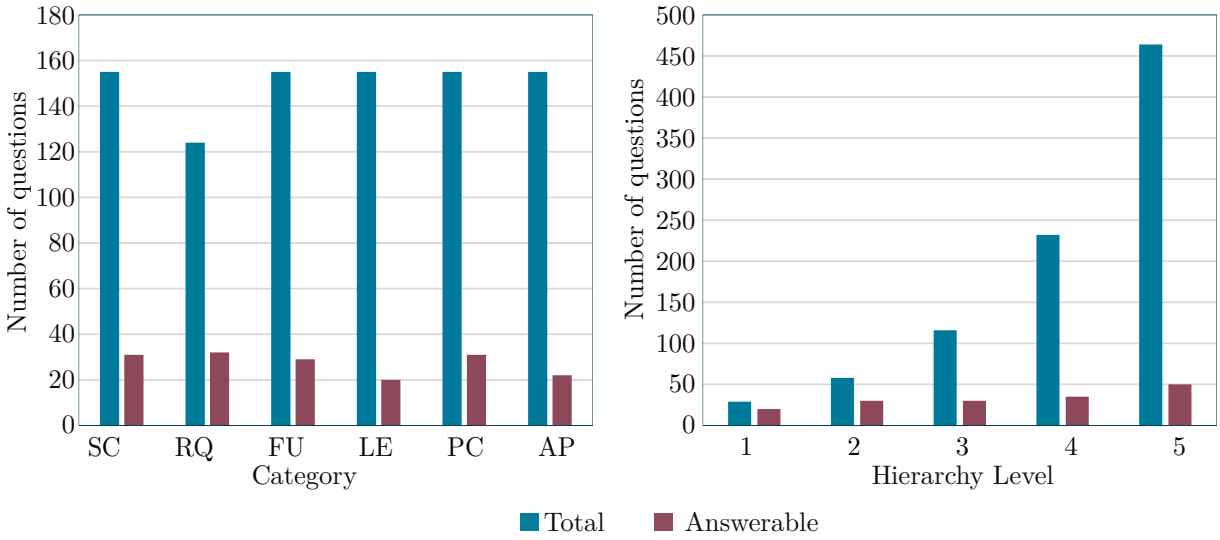

Figure 5.3.: Answered ECS questions by category (left) and by hierarchy level (right). Blue bars represent the total number of questions analyzed, while red bars denote the subset that could be answered with the information available in the database (yes)

The recursive questioning process, starting from 29 initial guiding questions, generated a total of 899 QA-pairs across five hierarchy levels with two subquestions per question. Of these, 165 questions could be answered using the available database, representing an 18.3% overall answer rate across all knowledge categories, as shown in Table 5.1 and Figure 5.3. It should be noted that one requirement-level guiding question was inadvertently omitted (4 instead of the intended 5), which resulted in 899 rather than 930 total QA-pairs. This minor deviation does not materially affect the subsequent analysis and phases.

Regarding the analysis by knowledge category, the distribution of answerable questions varies significantly. The questions regarding the requirements achieved the highest rela-

Table 5.1.: Overview of answered questions by category

Category	**Total Questions**	**Answered Questions**	**Answer Rate**	**Percentage of Total**
System Context (SC)	155	31	20.0%	16.5%
Requirements (RQ)	124	32	25.8%	13.2%
Functions (FU)	155	29	18.7%	16.5%
Logical Elements (LE)	155	20	12.9%	16.5%
Physical Components (PC)	155	31	20.0%	16.5%
Architecture Patterns (AP)	155	22	14.2%	16.5%
Total	**899**	**165**	**18.3%**	**100%**

tive answer rate at 25.8%. Despite the smaller number of total questions in this category (124 compared to 155 in the other categories), 32 questions were successfully answered, the highest absolute count across all categories (see Table 5.1). This suggests that system requirements were captured relatively well in the selected document corpus, although the overall coverage remains limited. The two categories, physical components (PC) and system context (SC), both achieved 20.0% answer rates, suggesting adequate documentation of concrete, technical aspects. Questions regarding functions (FU) achieved 18.7%, demonstrating reasonable coverage of operational behaviors. The categories architecture patterns (AP) and logical elements (LE) showed lower answer rates of 14.2% and 12.9%, respectively. These differences in answer rates reflect the varying representation of different types of knowledge within the specific document corpus used for this evaluation. The distribution indicates that the selected documents contain information across all six knowledge categories, ensuring that the subsequent knowledge synthesis can draw from diverse aspects of ECS systems.

The distribution across hierarchy levels shows the expected pattern of decreasing answer rates with increasing technical depth, as shown in Table 5.2 and the right diagram from Figure 5.3. Level 1 questions achieved 69.0% coverage, level 2 achieved 51.7%, and subsequent levels showed progressively lower rates (25.9%, 15.1%, and 10.8% for levels 3-5, respectively). This hierarchical distribution demonstrates that the document corpus provides strong foundational knowledge while containing progressively less detailed information at deeper technical levels. The pattern is consistent with the content of typical scientific papers, where fundamental concepts are more frequently discussed than concrete implementation details.

This statistical analysis provides valuable insights into the information content and distribution within the document corpus. The answer rates indicate which types of knowledge are well-represented in the available documents and which are less covered. The higher answer rate for requirements (25.8%) demonstrates that the documents contain substantial information about system specifications, while the lower rates for architecture patterns (14.2%) indicate less coverage of implicit design knowledge.

Table 5.2.: Overview of answered questions by hierarchy level

Hierarchy Level	**Total Questions**	**Answered Questions**	**Answer Rate**
Level 1 (Initial)	29	20	69.0%
Level 2	58	30	51.7%
Level 3	116	30	25.9%
Level 4	232	35	15.1%
Level 5	464	50	10.8%
Total	**899**	**165**	**18.4%**

The performance within these constraints provides valuable insights for the evaluation approach. The 165 successfully answered questions demonstrate that focused, high-quality responses can provide substantial value even from limited sources, emphasizing quality over quantity. The high answer rate of 69.0% at the first level ensures that essential system understanding is captured, which is critical for architectural reasoning. This constrained scenario creates an ideal test case for evaluating whether the RAG process can generate meaningful architectural insights from limited but focused information sources, presenting a valuable evaluation opportunity.

To illustrate the level of technical detail and contextual accuracy achieved by the recursive retrieval process, one representative QA-pair from the category *Architecture Patterns (AP)* is presented in Box 10.

Box 10: Example of a QA-pair at level 5

LLM-generated question:
AP.1.1.2.2.1 (Level 5): What are the specific control logic parameters and setpoint ranges used for the FCV to maintain bleed air pressure above the minimum threshold at the ECS pack interface, and how do these parameters interface with engine control systems to prevent compressor surge during transient operating conditions?

Answer (without thought process):
The Flow Control Valve (FCV) in the ECS bleed air supply system is controlled to maintain bleed air pressure at the ECS pack interface above a minimum threshold of approximately 25 psig, with typical operating pressures ranging from 30 to 45 psig during cruise conditions. The FCV control logic regulates the valve position to maintain pressure within ± 2 psig of the commanded setpoint, ensuring stable and sufficient bleed air flow to the ECS packs.

This control is achieved through a closed-loop system using pressure sensors and feedback mechanisms that continuously monitor bleed air pressure and adjust the FCV accordingly. The FCV interfaces directly with the engine's Full Authority Digital Engine Control (FADEC) system, coordinating bleed air extraction to prevent compressor surge during transient operating conditions. By modulating bleed air flow smoothly and maintaining pressure above the minimum threshold, the FCV helps preserve compressor surge margin and engine stability.

In summary, the FCV maintains bleed air pressure above 25 psig (minimum threshold), typically controlling within 30-45 psig, with a regulation precision of ± 2 psig. It works in coordination with engine control systems to modulate bleed air extraction dynamically, preventing compressor surge during transient events by avoiding abrupt pressure changes and ensuring stable airflow to the ECS packs [Ref: 617ef7af-02e1-40e8-9c2e-d6c6a902cdd9].

No further detailed numeric control parameters or algorithms are provided in the documents.

Phase 2 establishes the scope and coverage of the knowledge base that will be used in the subsequent synthesis report phase. The key questions for the next phase are therefore twofold: First, can the LLM effectively synthesize the 165 distributed answers derived from the conventional pneumatic ECS into a coherent and technically consistent system document? Second, how accurate, detailed, and technically valid is the resulting synthesis report?

5.4. Phase 3: Knowledge Synthesis Through LLM Integration

Phase 3 addresses exactly this question: can LLM-based synthesis consolidate fragmented QA-pairs into a coherent, RFLP*-structured document suitable for engineering use? The extraction provides complete source traceability but only as isolated QA-pairs. This phase, therefore, examines whether the QA-pairs can be consolidated into consistent technical reports that support subsequent architectural reasoning.

Following the modular prompting strategy evaluated in Study D (Section 4.2.4), the workflow shown in Figure 5.4 comprises three steps: prompt design, LLM-based synthesis, and systematic evaluation of quality and source fidelity.

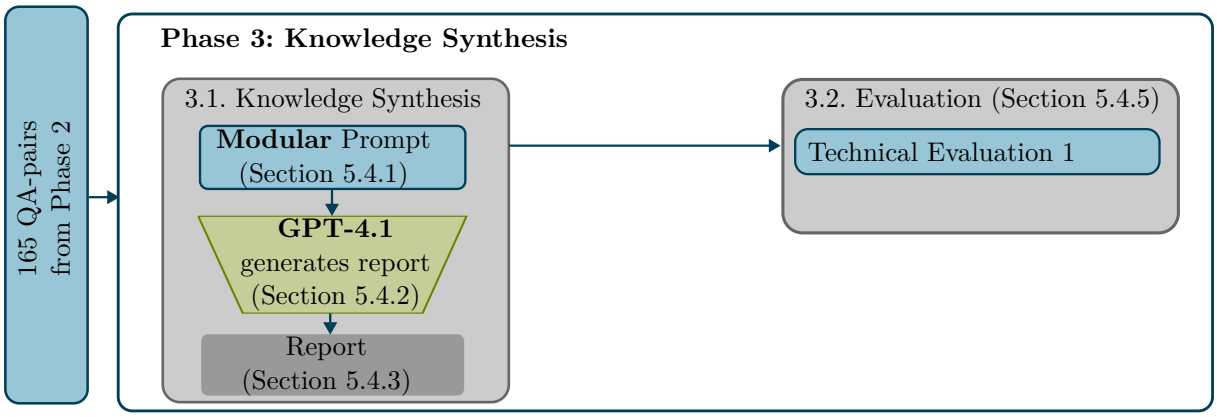

Figure 5.4.: Knowledge synthesis process

5.4.1. Prompt Engineering for Knowledge Synthesis

The knowledge synthesis process employs six specialized prompts, each building incrementally on previous results while following the extended RFLP* scheme. This modular approach ensures that the LLM can focus its full attention on each architectural domain while maintaining consistency across the complete document.

The system context analysis (Module 1) processes the 165 QA-pairs to establish fundamental system understanding through analysis of external interfaces, stakeholder relationships, and operational modes. The prompt used for Module 1 can be found in Appendix A.9 on page 198. The requirements extraction (Module 2) builds upon the system context to identify and structure functional and non-functional requirements. Module 3 (functional report) develops functional specifications focusing on functions and function flows. Module 4 (logical elements report) captures logical components, relationships, and architectural patterns, while Module 5 (physical components report) specifies components, interfaces, and hardware implementation details. Finally, the architectural views report (Module 6) summarizes all previous analyses and presents the overall system architecture from various perspectives. Each module progressively integrated the original QA-pairs and all previously created reports.

Each module followed a standardized prompt structure to ensure methodological consistency and comparability of results. The general prompt framework comprised the following elements:

- Task Definition: "*Create a comprehensive report on the Module X of the Environmental Control System (ECS) based on the provided documents.*"
  The prompt explicitly described the engineering domain, the expected report scope, and its dependency on previously generated modules.

- Output Format Specification: The report was required in structured HTML format with a comprehensive table of contents. Tables and lists were to be used instead of diagrams to ensure readability and comparability across all modules.

- Transparency and Color-Coding System: A uniform legend indicated the origin of information and distinguished between content directly extracted from QA-pairs, information derived from multiple sources, elements inferred from previous documents, and assumed information with explicit justification.

- Document Structure Definition: Each prompt predefined a standardized section layout (e.g., overview, detailed specifications, interfaces, traceability matrices) aligned with the RFLP* logic.
- Quality and Traceability Goals: Each report was required to be comprehensive, consistent, verifiable, and fully traceable to source QA-pairs or previous module IDs. Missing information had to be explicitly flagged as assumptions.

5.4.2. LLM Configuration and Settings

GPT-4.1 was used as the LLM, as specified in Study E (Section 4.2.5). This model showed optimal performance in several evaluation dimensions, including factuality, technical precision, source fidelity, and completeness, while maintaining computational efficiency. The temperature parameter was configured to 0.2. This setting enables structured knowledge synthesis and organizational creativity while promoting greater adherence to source material and reducing the sampling variability that can lead to factual deviations. The configuration thus balances response predictability with the creative flexibility necessary for effective knowledge integration and synthesis of architectural documentation.

5.4.3. Knowledge Synthesis Report Generation

After all six modular reports had been completed, a rule-based consolidation process was implemented to combine the individual documents into a single, comprehensive technical HTML report. This technical process involved merging the text content and adapting the table of contents and hyperlink functionalities for the consolidated document. The original content, formatting, and three-level color coding system remained unchanged. As the consolidated report exceeds 48 pages[2], excerpts are provided below to illustrate its structure and content quality. The examples presented here primarily demonstrate the format and appearance of LLM-generated tables and structured content. A more comprehensive selection of report sections can be found in Appendix A.10 on page 204.

Table 5.3 demonstrates the systematic identification of external system interfaces, showing how the LLM identified all major aircraft systems that interact with the ECS. The consistent green coding indicates that all interface descriptions were directly traceable to source QA-pairs, demonstrating high source fidelity. The table thus presents the complete set of primary external interfaces identified by the LLM through the given prompt and QA-pairs, with the "Source" column linking each entry to the corresponding QA-pair (e.g., ID "SC.1").

Table 5.3.: ECS external system interfaces (example of knowledge synthesis report)

External System	Interface Description	Source
Engine/APU Bleed Air System	Provides high-pressure, high-temperature air to ECS packs for air conditioning, pressurization, and ventilation	SC.1
Ram Air System	Supplies ambient cooling air to ECS heat exchangers for heat rejection	SC.1
Aircraft Electrical System	Powers ECS controls, valves, fans, and sensors	SC.1
Cabin Pressurization System	Interacts via outflow valve to regulate cabin pressure	SC.1

[2]The page count was determined by exporting the original HTML document via the browser's print function ("*Save as PDF*") using standard A4 settings and counting the resulting PDF pages.

Table 5.4 displays four of the 13 functional requirements identified by the LLM from the vector database, chosen as representative examples. The quality of the requirement specification is notable, with clear descriptions and precise quantitative parameters (e.g., temperature ranges, efficiency thresholds, response times). The consistent green transparency coding demonstrates that these requirements were systematically derived from the source QA-pairs rather than assumed, indicating traceability to the vector database. A complete list of all 13 identified functional requirements can be found in the Appendix in Table AT.11[3] on page 209.

Table 5.4.: Key ECS functional requirements (example of knowledge synthesis report)

ID	Title	Description	Transparency
FR-001	Cabin Temperature Control	The ECS shall provide temperature-controlled air to the flight deck, passenger cabin, and cargo compartments, maintaining setpoint temperatures between 18-24°C (64-75°F) under all operational conditions.	[FU.1, FU.1.1, RQ.4]
FR-002	Cabin Pressurization Control	The ECS shall maintain cabin pressure equivalent to an altitude of 6,000-8,000 ft, with a maximum pressure differential of 8-9 psi, throughout all flight phases.	[FU.1, FU.1.1, SC.2]
FR-003	Cabin Humidity Control	The ECS shall control cabin humidity by removing moisture from supply air, maintaining water separator efficiency above 85% and preventing condensation or ice formation in the cabin and ECS components.	[FU.1.1, SC.4.2.1.2]
FR-008	Health Monitoring	The ECS shall continuously monitor system health using sensors (temperature, pressure, humidity, airflow, valve position, turbine speed) and provide early fault detection, isolation, and reporting to maintenance personnel.	[FU.5, FU.5.1, SC.3.1]

Table 5.5 shows the complete functional hierarchy containing 11 functions. The systematic decomposition from the main function (FN-001) to primary functions demonstrates proper systems engineering methodology. The classification structure ("Main" and "Primary") follows established functional architecture principles, while the source citations show comprehensive coverage across different QA-pairs. This hierarchical approach enables clear traceability from high-level system purposes to specific operational behaviors.

Table 5.5.: ECS function hierarchy (example from knowledge synthesis report)

Function ID	Name	Classification	Source
FN-001	Cabin Environmental Control	Main	FU.1
FN-002	Temperature Control	Primary	FU.1.1, FU.1.1.1
FN-003	Pressurization Control	Primary	FU.1.1
FN-004	Humidity and Moisture Control	Primary	FU.1.1, FU.1.1.1
FN-005	Ventilation and Air Quality	Primary	FU.1, FU.3.2.1, SC.3.2.2
FN-006	Cabin Air Distribution	Primary	FU.2, RQ.1.1.2
FN-007	System Monitoring and Diagnostics	Primary	FU.2, FU.5, FU.5.1
FN-008	Transition and Mode Management	Primary	FU.3
FN-009	Safety and Protection	Primary	FU.4
FN-010	Redundancy and Emergency Operation	Primary	FU.4.2, A.3
FN-011	Subsystem Integration and Control	Primary	LE.1, LE.3, LE.5

[3]Appendix elements use distinct prefixes: AT for tables, AF for figures, and AE for equations to prevent reference ambiguity.

These examples illustrate the systematic structure and professional quality achieved by the LLM-generated knowledge synthesis report. The consistent application of the transparency system, precise technical specifications, and comprehensive source citations demonstrates that the methodology successfully transforms dispersed QA-pairs into a structured, engineering-grade document that maintains clear provenance throughout all architectural domains.

The consolidated report demonstrates coverage across all architectural domains:

- Total Document Length: 48 pages
- Requirements Coverage: 13 functional and 15 non-functional requirements
- Functions: 11 main functions with complete hierarchical decomposition
- Logical Elements: 9 logical subsystems with detailed interface specifications
- Physical Elements: 32 components with full specifications and relationships
- Architectural Views: 8 distinct viewpoints providing comprehensive system perspectives

These examples illustrate the systematic structure and quality achieved by the LLM-generated knowledge synthesis report. The consistent application of the transparency system, precise technical specifications, and clear referencing of each information element to its originating QA-pair demonstrate that the methodology successfully transforms dispersed knowledge into a structured document with full traceability. Overall, the synthesized report presents a logically coherent and technically plausible depiction of the ECS. The generated content shows a high degree of internal consistency, no indications of hallucination, and a stable, technically sound layout. The resulting document provides a convincing and comprehensive overview of the system's architecture and functionality, closely reflecting the characteristics of an actual ECS.

Although the report appears comprehensive and well-structured, it must be emphasized that it was generated entirely through an LLM. Such outputs may still contain hallucinations, misinterpretations, or incorrect attributions, and their non-deterministic nature means that results can vary between executions in terms of phrasing, organization, or emphasis. This inherent variability challenges consistent evaluation and necessitates systematic verification of source fidelity, confidence classification, and logical consistency.

5.4.4. Evaluation of Knowledge Synthesis Report

The knowledge synthesis report demonstrates that LLMs can generate a comprehensive technical document from extracted QA-pairs. This capability, however, requires an evaluation to determine whether the synthesized knowledge can reliably serve as a foundation for subsequent architecture generation. The evaluation, therefore, addresses three questions: (i) does the LLM accurately interpret and integrate information from source QA-pairs, (ii) are citations traceable back to their sources, and (iii) are the established links between elements (requirements, functions, logical and physical components) consistent and traceable across the report?

The evaluation framework follows the methodology established in Section 3.3.2, examining three critical dimensions that determine whether the synthesized document meets engineering-grade standards for subsequent development:

1. Source fidelity: An assessment of whether the LLM has correctly assigned source references to each statement in the synthesized document. This evaluation confirms that the claims made in the generated text can be substantiated in the referenced source materials (165 QA-pairs), ensuring accurate citation and proper attribution of information to its sources.
2. Confidence Classification Accuracy: Analysis of whether the three-tier confidence classification system (green/orange/red) correctly reflects uncertainty and derivation status of each information element. Green indicates direct extraction from sources, orange represents logical inferences and derivations, while red flags assumptions or speculative content where source support is limited or absent.
3. Traceability Assessment: Evaluation of whether relationships across requirements, functions, logical elements, and components are mapped consistently

Source Fidelity Analysis

The evaluation assesses whether synthesized statements represent the underlying knowledge base as trustworthy or whether the LLM generates hallucinated content. It focuses on green-classified statements marked as "*directly traceable to source content*". Each claim from the synthesized ECS report (in this run $n = 183$, varying across synthesis instances) is systematically checked against its QA-pair source to verify accuracy and fidelity. Details and full results are provided in Appendix A.11.

A claim corresponds to a specific statement in the synthesized report. For instance, Table 5.3 lists the external interface "*Engine/APU Bleed Air System*" with the description "*Provides high-pressure, high-temperature air to ECS packs for air conditioning, pressurization, and ventilation*". This is traced back to QA-pair SC.1, which states (excerpt): "*The ECS physically interfaces externally with the aircraft's engine and APU bleed air systems, providing high-pressure hot air, ... and with the ram air system, the aircraft electrical system, the cabin pressurization system, and the water separation and drainage systems.*" Since the synthesized claim corresponds directly to this excerpt, it is classified as correct (direct source alignment). If no such match exists, the claim is flagged as value-added or divergent. This procedure applies systematically to all claims.

The evaluation reveals a high degree of fidelity between synthesized content and source material. The results demonstrate that:

- 91.8% of statements align directly with their sources, indicating reliable knowledge transfer.
- 7.1% are value-added statements, containing technically correct information that was not found in the original source content. Further analysis revealed that these values or statements were present in other sources within the corpus, but had not been attributed to them.
- 1.1% showed source divergence, representing technically plausible assertions that were not substantiated in the referenced sources.

The predominance of aligned statements provides strong evidence for the reliability of the synthesis process. Value-added cases reflect additional specifications beyond the explicit source content. For example, the requirement FR-002 in Table 5.4 specifies a cabin pressure differential of "*8-9 psi*", although the cited QA-pair only states that cabin altitude shall not exceed 8,000 ft. While technically correct and supported by other QA-pairs, this detail is not present in the referenced source and is therefore classified as value-added. Divergent cases are rarer (1.1%), but more critical. They introduce technically plausible assertions that are not substantiated in the cited material, such as a safety

requirement formulated in terms of regulatory compliance without explicit support in the QA-pair. For instance, the synthesized report states that "*the ECS shall be designed to meet safety requirements for catastrophic and hazardous failure conditions as defined by applicable aviation regulations (e.g., FAA/EASA)*", although the corresponding QA-pair only describes specific operational safety thresholds (e.g., temperature, pressure, flow, and monitoring limits) without mentioning certification standards. The risk in both cases (value-added and divergent) lies in their green classification, which falsely suggests direct source support and could lead to misinformed safety-critical reasoning.

In summary, the source fidelity evaluation demonstrates that LLM-based synthesis achieves high technical accuracy (91.8% aligned, 7.1% value-added, 1.1% divergent) and can therefore be considered highly reliable for applications in aircraft engineering. The main risk lies in misclassified confidence levels. To address this, future work should implement automated claim-level verification. A practical approach would be to extract synthesized statements, map their source references, and compare them automatically against the QA-pairs. An additional evaluation loop could then reassess green classifications and adjust them where necessary. Such automation would enable scalable validation for larger knowledge bases, reduce the risk of misclassification, and maintain consistency across datasets. Overall, LLM-based synthesis shows strong potential for aircraft documentation, provided that confidence classification systems are validated and verification workflows are systematically integrated.

Confidence Classification Assessment

The three-tier visual coding scheme is evaluated in terms of its ability to communicate information reliability and highlight knowledge gaps. The complete analysis is provided in Appendix A.11 on page 218. The confidence classification employs a color-coded system, where green indicates direct source traceability, orange represents derived information from multiple sources, and red highlights assumptions or explicit knowledge gaps.

This assessment is closely linked to the results of the source fidelity analysis, as the reliability of each color category ultimately depends on whether statements are accurately grounded in their referenced QA-pairs. In this sense, the confidence classification does not constitute an independent evaluation but rather a secondary interpretation of source fidelity from a user-oriented perspective, focusing on how effectively the visual system communicates information certainty to engineers.

Green-classified statements prove highly reliable. More than 90% align directly with their sources. This confirms that green consistently signals source-grounded information, enabling engineers to distinguish between verified technical data and synthesized assertions with high confidence. For example, the requirement to maintain cabin pressure between 6,000 and 8,000 ft is marked green because it appears explicitly in the QA-pairs.

The red classification statements are particularly valuable in explicitly identifying critical knowledge gaps and specification limitations within the source documents. Representative examples include statements highlighting unavailable material specifications, undefined interface mechanisms, and incomplete user interface flows for emergency operations. These red classifications serve as explicit risk indicators, highlighting areas where assumptions have been made or where critical technical specifications are lacking in the knowledge base. This transparent identification of knowledge limitations enables engineers to prioritize additional research and specification development activities.

Despite the overall effectiveness of the classification system, some inconsistencies were observed where statements containing knowledge gaps or assumptions were classified as green instead of orange or red. This indicates opportunities for improving classification accuracy, particularly in distinguishing directly traceable information from statements

with embedded uncertainties. However, the systematic presentation of information uncertainties through red classifications provides significant engineering value by enabling risk-informed design decisions and proactive identification of technical risks.

Traceability Assessment

The traceability assessment applies a systematic approach to evaluate consistency across multiple abstraction layers. Established SE standards such as ISO 15288 [11] and the INCOSE Systems Engineering Handbook [12] require that system elements remain traceable to each other. The evaluation, therefore, examines whether this requirement is fulfilled in the synthesized document. A hierarchical model is used, spanning requirements, functions, logical elements, and components. Each relationship is reviewed using a three-state classification: correct, incorrect, or missing. The complete results are shown in Table 5.6 and further details are provided in Appendix A.11 on page 219.

Table 5.6.: Overall traceability assessment results

Traceability Category	**∑ Mappings**	**Correct Mappings**	**Incorrect Mappings**	**Missing Mappings**
Requirements → Functions	22	22	0	0
Functions → Logical Elements	13	13	0	0
Logical Elements → Components	12	12	0	0
∑	**47**	**47**	**0**	**0**

The assessment covers 47 individual mappings across three architectural layers. All mappings are correct, resulting in 100% accuracy. This outcome represents a significant achievement in automated document generation, demonstrating that the synthesized content maintains both structural and semantic consistency. Each requirement is correctly linked to its implementing functions, which in turn connect to logical elements and then to physical components. For example, a temperature control requirement is traced through the corresponding thermal management functions and mapped to heat exchanger components.

This level of traceability enables seamless integration into established systems engineering workflows such as change impact analysis, verification planning, and requirements validation. Achieving perfect consistency across all mappings also demonstrates the scalability of the approach, suggesting that larger and more complicated systems can benefit from similar automated methods while maintaining reliable traceability.

Conclusion of the Evaluation

The evaluation shows that LLM-assisted knowledge synthesis achieves a high accuracy suitable for aircraft system design. The synthesized document maintains high source fidelity (91.8%), communicates uncertainty through confidence classification, and preserves complete traceability across abstraction levels (100%). These results confirm that the methodology can transform fragmented QA-pairs into a reliable technical document while reducing manual effort.

It is important to note that the quantitative metrics presented in this evaluation (91.8% source fidelity, 100% traceability, specific utilization patterns) reflect the assessment of one particular synthesis report instance generated under defined conditions (GPT-4.1 model,

temperature 0.2). Because LLM outputs are inherently non-deterministic, repeated executions of the synthesis process may yield reports with varying characteristics, which can result in differences in evaluation metrics. While the evaluation methodology and quality assessment framework remain transferable and replicable, the specific numerical results should be interpreted as representative of this synthesis instance rather than universal guarantees for all future applications.

Overall, Phase 3 demonstrates that the methodology delivers highly reliable results. The knowledge synthesis produced an engineering-grade document with high accuracy, complete traceability, and effective confidence communication. This performance establishes a solid foundation for architectural reasoning and confirms that the methodology performs well up to the level of structured documentation and component analysis.

5.5. Phase 4: Architecture Generation Through LLM Reasoning

Phase 4 builds directly on the synthesized document and addresses the transition from analysis to design, as can be seen in Figure 5.5. The first step applies a structured Reuse/Adapt/New analysis of ECS components to determine which elements can be carried over, which require modification, and which must be newly developed for ambient air operation. This provides the methodological bridge toward the second step, where LLMs are challenged to generate complete system architectures. The evaluation, therefore, examines not only whether the methodology can classify and adapt existing knowledge but also whether it can extend this reasoning to coherent system-level designs that align with established engineering principles. It also reveals the current boundaries of GenAI-assisted engineering design.

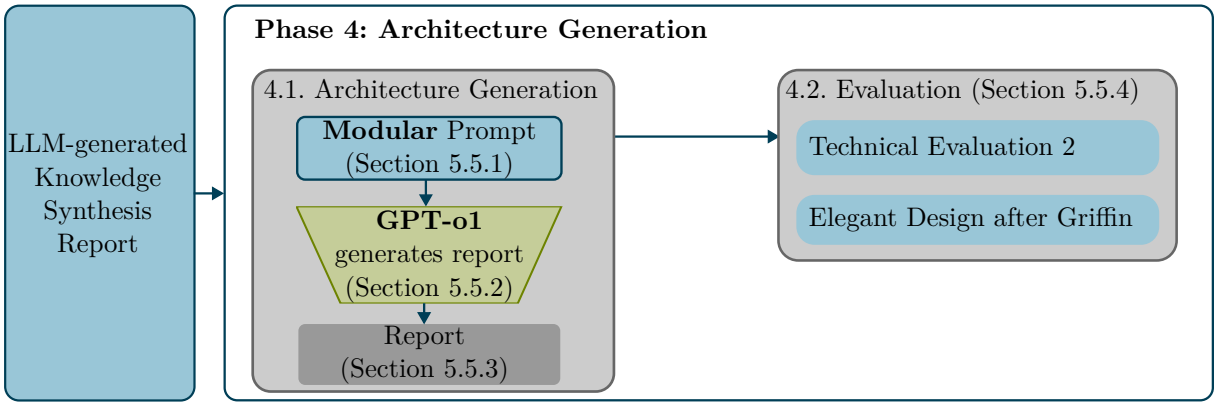

Figure 5.5.: Architecture generation process of the eECS

5.5.1. Prompt Engineering for Architecture Generation

Phase 4 follows a four-stage prompting strategy designed to transform existing ECS knowledge into new concepts based on new boundary conditions. Unlike Phase 3, which worked with fragmented QA-pairs, this phase utilizes the consolidated knowledge synthesis report to enable systematic reasoning.

- Prompt 1 - Component Reusability Analysis: The first prompt focuses exclusively on analyzing existing ECS components for potential reuse, modification, or replacement in an ambient air-based system[4]. This analysis provides the foundation for subsequent architectural decisions by systematically evaluating the compatibility of each component with the new ambient air boundary condition.
- Prompt 2 - RFLP Redefinition: The extended RFLP* scheme is applied to redefine system boundaries, requirements, and logical structures for ambient air intake.
- Prompt 3 - First Architecture Development: A complete system architecture is generated, integrating all prior analyses into a coherent design. This prompt includes embedded calculation rules and JavaScript-based computational instructions to guide the model in evaluating thermodynamic performance parameters and component sizing.
- Prompt 4 - Alternative Architecture and Comparison: A second, conceptually distinct design is created and systematically compared to the first through trade-off analysis of performance, complexity, cost, and risk. Similar to Prompt 3, JavaScript-based calculation scripts were included to make mathematical operations possible within the generation process, allowing the model to compute thermodynamic parameters and component performance directly.

To support this process, the classification system used in Phase 3 is extended from three to four levels. Green marks elements reused without change. Orange indicates modifications of existing elements. Blue denotes entirely new elements. Red highlights assumptions or unresolved aspects requiring further specification. This finer granularity improves prioritization of development effort and risk awareness in early design phases.

Following the evaluated approach from knowledge synthesis (Section 5.4.1), each prompt builds incrementally on the previous results. Prompt 1 receives the complete knowledge synthesis report as primary input, ensuring that component reusability analysis is grounded in comprehensive system understanding. Prompt 2 integrates both the knowledge synthesis report and the component reusability analysis, enabling informed RFLP redefinition based on component-level insights. And Prompt 3 incorporates all previous analyses plus additional technical requirements, facilitating comprehensive architecture development with full contextual awareness. As an example, Prompt 3 can be found in Appendix A.9 on page 199.

Each architecture generation prompt followed a standardized framework to ensure consistency and reproducibility across all four modules. The general structure included the following elements:

- Task Definition: "*Develop a complete architectural specification for the Ambient Air Intake ECS based on the provided documents and analysis results.*"
  Each prompt defined the architectural scope, design stage, and intended degree of modification or innovation.
- Input Documents: All prompts referenced outputs from previous phases, including the knowledge synthesis report, to ensure full traceability across development stages.
- Boundary Conditions and Target Specifications: The prompts included predefined environmental and performance targets reflecting the operational limits of an ambient air intake ECS: minimum air flow of 0.55 kg/min per passenger, cabin temperature control between 18°C and 30°C, cabin pressure equivalent to a maximum

[4]The term *ambient air-based ECS* refers to the eECS concept similar to that used in the Boeing787, where the supply air is drawn directly from the surrounding atmosphere at the aircraft nose rather than extracted as bleed air from the engines. This configuration eliminates the traditional bleed-air dependency and introduces new thermodynamic and integration challenges.

altitude of 8,000 ft, and intake air conditions ranging from -60°C to +50°C at 0.2-0.3 bar during cruise flight.

- Output Format: Structured HTML format with tables, cross-references, and a color-coded transparency system ensuring clarity and comparability.
- Calculation Rule: Embedded JavaScript-based computation rules enable thermodynamic and power calculations, supporting pressure, temperature, and performance verification during generation.
- Color-Coding System: Four-level classification: green (reused components), orange (modified), blue (new), red (assumed or unspecified).
- Quality and Traceability Goals: Emphasis on engineering realism, transparent reasoning, and consistent linkage from requirements to architectural solutions.

For the first time, JavaScript-based calculation routines were embedded directly into the prompts to enable explicit thermodynamic and power evaluations. This integration allowed step-by-step performance calculations within the generation process, extending the prompting approach from descriptive synthesis to semi-analytical engineering reasoning. The complete prompt, including the embedded JavaScript-based computation routines, is provided in Appendix A.9 on page 199.

5.5.2. LLM Configuration and Settings

For the architecture generation phase, OpenAI's GPT-o1 model was selected based on the significantly reduced input token requirements compared to the knowledge synthesis phase. The architecture generation prompts, while comprehensive, require substantially fewer input tokens than the complete Q&A database used in knowledge synthesis, making the GPT-o1 reasoning model computationally feasible for this application.

The GPT-o1 model family represents a paradigm shift in LLMs, employing chain-of-thought reasoning as a fundamental processing mechanism rather than an optional enhancement. This reasoning capability is particularly valuable for architecture generation, where complex technical relationships, trade-offs, and design decisions require systematic analytical thinking that extends beyond pattern matching.

Unlike traditional LLMs, reasoning models operate without user-adjustable temperature parameters, as their reasoning process inherently balances accuracy and creativity. The model's internal reasoning mechanism automatically modulates between analytical rigor and innovative thinking based on the specific requirements of each architectural task.

5.5.3. Architecture Generation Report

Following the completion of all four prompts, a systematic consolidation process was implemented to integrate the individual HTML reports into a comprehensive technical document. This process employed the same rule-based methodology evaluated in the knowledge synthesis phase (Section 5.4.3), ensuring consistency in document standards and functionality.

The consolidated architecture generation report demonstrates the capability of the chosen LLM model to develop technical solutions through systematic engineering processes. Unlike the knowledge synthesis phase, which focused on extracting and organizing existing information, the architecture generation phase requires problem-solving. To illustrate the quality and depth of the generated content, the following examples show key outputs from

the architecture development process. As the full report cannot be presented in detail here, its key characteristics are summarized below:

- Length: 32-page consolidated report
- Subsystems/Components: 7 identified elements classified as Reuse, Adapt, or New development
- External Boundaries: 7 external interfaces systematically redefined
- Stakeholders: 5 groups (Flight Crew, Maintenance Personnel, Passengers, Regulatory Entities, Aircraft OEM)
- Requirements: 10 functional, 4 performance, 4 interface, 4 safety
- Functions: 8 core functions identified
- Logical Elements: 6 elements, including ECS Controller, Distribution Subsystem, etc.
- Architectures: 2 alternative proposals with full stage-by-stage analysis

The following tables and figures highlight representative examples from this report, beginning with the component reusability matrix.

Table 5.7 bridges the step from knowledge synthesis to architecture work by showing which conventional ECS elements can be carried over and which must change for ambient-air operation. The color coding follows a simple logic. The green background in the first three columns indicates items that originate from the conventional ECS and can, in principle, be reused. The column "Reusability Status" in the table states the action, such as direct reuse, minor adaptation, or major modification. Orange entries indicate the need to adapt hardware or software, such as valve trim ranges or control algorithms. Red entries mark elements that must be introduced as new, with the electrically driven compressor as the

Table 5.7.: Excerpt of the component reusability matrix from the report

Comp. ID CMP-	Comp. Name	Current Function	Reusability Status	Required Modification
HW-001	Air Cycle Machine (ACM)	Core air conditioning using bleed air expansion	MAJOR Modification	Replace or redesign compressor/-turbine stages to handle lower pressure and different temperature
HW-012	Flow Control Valve (FCV)	Regulates mass flow of bleed air	MINOR Adaptation	Valve trim/actuator settings updated for new pressure ranges
HW-022	Mixing Manifold	Combines condit. air with recirculated air	DIRECT REUSE	No modifications needed (manifold design suitable for new supply conditions)
HW-010	Water Extractor	Removes moisture, >85% efficiency	MINOR Adaptation	Recheck capacity for lower inlet temperature; may need minor resizing or new coalescer element
SW-001	ECS Controller Unit	Controls ECS modes, monitors sensors	MAJOR Modification (Software)	New control algorithms for compressor speed, surge margin with ambient intake airflow
HW-025	Outflow Valve	Regulates cabin pressure via CPCS	DIRECT REUSE	No major hardware change required; same function
HW-031	Electrically Driven Compressor	Provides compressed air for ECS	REPLACEM./ NEW COMPONENT	Full sizing for ambient intake flow; HPC design for 3-4x compression ratio depends on flight altitude

key example. A few cells exhibit minor inconsistencies in background color. However, the status column determines the final classification and should be considered decisive. Based on this structured reuse, adapt, and new analysis, the following exhibits extend the design toward system boundaries and complete architectures.

Table 5.8 demonstrates the systematic redefinition of system boundaries for the ambient air-based architecture. The color coding clearly distinguishes between unchanged systems (green), modified interfaces (orange), and completely new elements (blue). This boundary analysis represents a critical step, demonstrating how the LLM accurately identified that eliminating bleed air would necessitate new ambient air intake systems while maintaining existing interfaces where possible.

Table 5.8.: Excerpt of the external system boundary definition from the report

External System	System Type	Interaction Description	Interface Requirements	Depend. Level	Change from Original ECS
Fore Fuselage Ambient Inlet	Ambient Air Source	Supplies ∼0.2 - 0.3 bar external air for ECS compression	Must accommodate icing protection, intake sensor data	High	NEW (Ambient intake replaced bleed source)
Engine/APU Bleed System	Pneumatic Source	Now used as optional or backup source if dual-mode is retained	Reduced or standby interface if ambient compression is primary	Low (in Ambient-only configurat.)	MODIFIED (was primary, now second./ backup)
Ram Air System	Cooling Air Source	Cool ECS heat exchangers with ambient flow at variable altitude	Valves, fans remain as before for heat rejection	Medium	UNCHANGED
Aircraft Electrical System	Electrical Power	Supplies power to the newly introduced HPC (High-Pressure Compressor) or electric compressor unit	Must handle HPC start-up loads plus steady-state ∼10 - 15 kW (approx.)	High for HPC-driven ECS	NEW or significantly expanded draw
Cabin Pressurization System	Pneumatic, Mechanical	Regulates cabin outflow; interacts with ECS to maintain 0.75 bar cabin supply at altitude	Outflow valve logic unchanged	High	UNCHANGED

Tables 5.9 and 5.10 illustrate the two distinct architectural approaches generated by the LLM. Architecture 1 uses a single-stage high-pressure compressor (HPC) design prioritizing simplicity, while Architecture 2 employs a two-stage HPC design with intercooling to manage thermal loads. Both tables demonstrate the capability of the LLM for detailed thermodynamic analysis, including pressure and temperature calculations at each stage. The mathematical verification columns demonstrate the correct application of compression formulas, while the design reasoning columns reveal the engineering decision-making processes.

The progression from Architecture 1's initial design problem illustrates the LLM's engineering problem-solving capabilities. The system first generated a single-stage HPC design that achieved the target compression ratio but produced a final manifold pressure of only 0.76 bar, falling short of the required $\geq$0.80 bar specification. Critically, the LLM independently identified this shortfall and conducted a systematic root cause analysis, determining that the HPC compression ratio of ∼4.0 was insufficient once downstream pressure losses (∼0.20 - 0.24 bar total) were properly accounted for.

Table 5.9.: Architecture 1: single-stage HPC component analysis (representative sample)

Stg.	Component	Input P (bar)	Input T (°C)	Output P (bar)	Output T (°C)	ΔP (bar)	Mathematical Verification	Design Reasoning
1	HPC	0.25	−50	1.00	∼ 130	+0.75	Verified with analysis tool	Single stage HPC, discovered final manifold < 0.8 bar
2	Large Final Cooler	1.00	130	0.92	∼ 30	−0.08	Pressure drop = 0.08 bar; Heat rejection to ram air	Cool air to ∼ 30°C feasible with large cooler
3	Water Separator	0.92	30	0.90	25-28	−0.02	Minor pressure drop, dryness ∼ 85 - 90%	Same design from original ECS
4	Mixing Manifold	0.90	25-28	0.88	25	−0.02	Manifold mixing los-ses	Reused from original ECS, minimal changes
5	(ducts, valves, filters, margin)	0.88	25	∼ 0.76	∼ 25	−0.12	Sum of 0.04 (duct), 0.03 (valves), 0.02 (HEPA), 0.03 (safety marg.)	We only get 0.76 bar, short of 0.80 bar spec

Table 5.10.: Architecture 2: two-stage HPC component analysis (representative sample)

Stg.	Component	Input P (bar)	Input T (°C)	Output P (bar)	Output T (°C)	ΔP (bar)	Mathematical Verification	Design Reasoning
1	HPC Stage 1	0.25	−50	0.50	−60	+0.25	*Equation in report*	First-stg. compr. cut in half to keep temp. mod.
2	Intercooler	0.50	−60	∼ 0.47	25	−0.03	Removes 35 K, pressure drop −0.03 bar	Inlet to HPC Stg. 2 −25°C improves efficiency
3	HPC Stage 2	0.47	25	1.05	∼ 200	+0.58	*Equation in report*	Second-stg. completes compr. to ∼ 1.05 bar
4	Large Final Cooler	1.05	200	0.97	30	−0.08	Final drop −0.08 bar, air cooled ∼ 170 K down	Similar cooler design to Architecture 1, sized for 1.8 kg/s
5	Water Separator	0.97	30	0.95	25-28	−0.02	Removes condensate at final ∼ 30°C	Directly reused from original ECS
6	Mixing Manifold	0.95	25-28	0.93	25	−0.02	Mixing losses −0.01 - 0.02 bar	Reused manifold design, no big changes
7	(Ducts, Valves, Filters, margin)	0.93	25	∼ 0.81	∼ 25	−0.12	Sum of 0.04 (duct) + 0.03 (valves) + 0.02 (HEPA) + 0.03 (safety marg.)	Net ≥ 0.8 bar at manifold meets specification

The system then performed design iteration, recognizing that while the excess HPC exit temperature (∼130°C) remained manageable with appropriate cooling, a higher discharge pressure margin was required. This led to the development of Architecture 2's two-stage approach with intercooling, which successfully achieved the pressure specification

while maintaining temperature within acceptable bounds. This autonomous problem identification, root cause analysis, and iterative design refinement demonstrate the LLM's capacity for engineering problem-solving that mirrors the thought processes of experienced system architects, a critical capability for practical engineering applications.

Figure 5.6 shows the two generated architectures and the direct comparison to the eECS of the B787. Architecture 1 implements a single high-pressure compressor (HPC) that compresses ambient air from ∼0.25 bar to 1.0 bar, followed by a heat exchanger, water

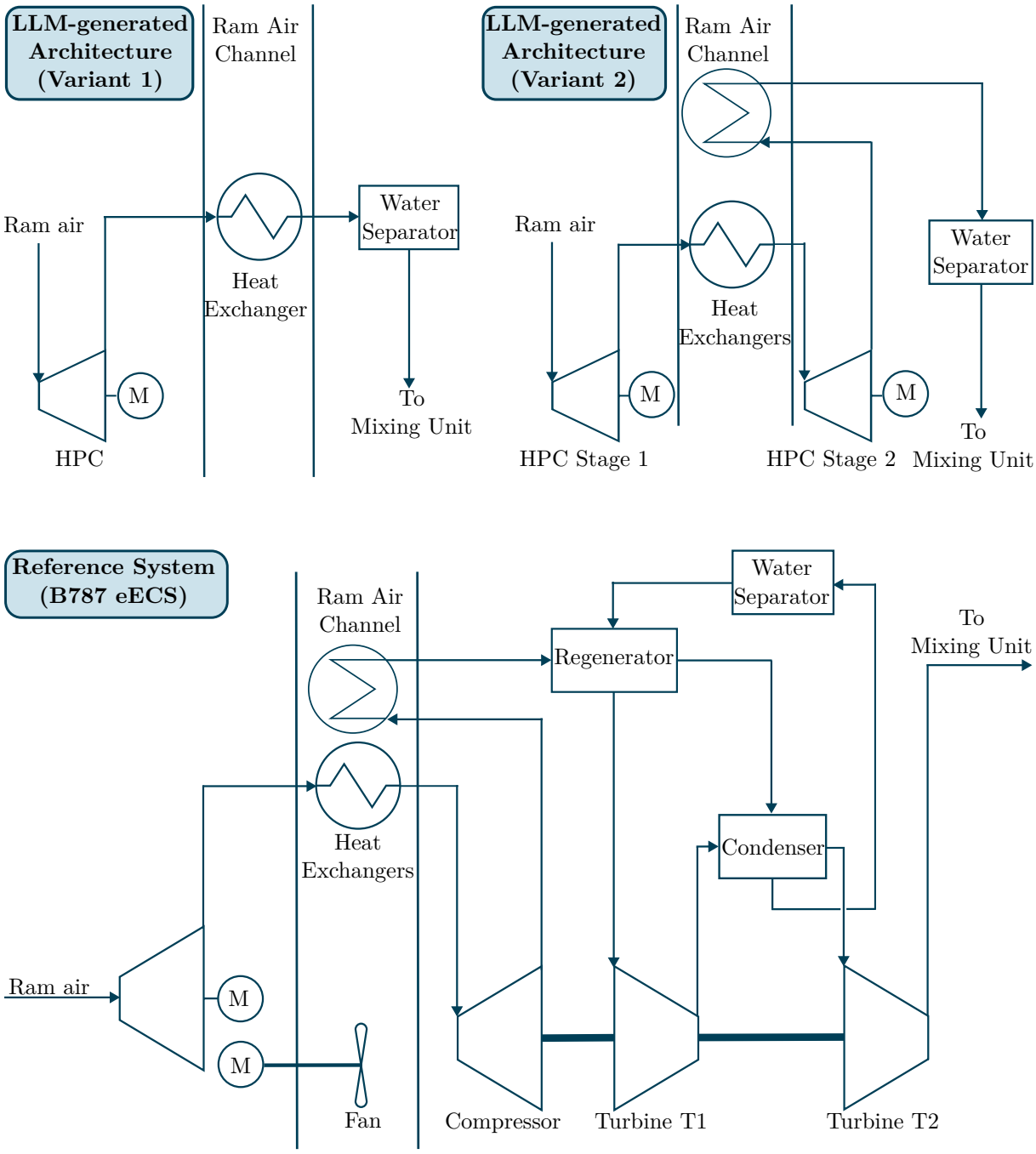

Figure 5.6.: System architecture of variant 1 (left), variant 2 (right), and the reference system of the B787 (bottom)

separator, and the mixing unit. However, according to the bilancing calculations, this simple variant only achieves 0.76 bar at the final distributor, failing to meet the required specification of $\geq$ 0.80 bar. Architecture 2 addresses this issue by using a two-stage compressor architecture comprising HPC Stages 1 and 2, which are connected by an intercooler for intermediate cooling and a water separator. This more complicated solution assumes 1.05 bar after the second stage, successfully meeting the required pressure specification of 0.81 bar. Both architectures represent a system philosophy that does not rely on engine bleed air. Instead, they use electrically driven compressors to compress ambient air. It should be noted that the schematic diagrams were not created by the LLM, as current LLMs have limitations in generating complex technical schematics with precise component representations and proper engineering symbology, but serve to visualize the architectures described in the tables.[5]

Due to the comprehensive nature of the architecture generation report, selected representative sections are included in Appendix A.12 on page 222 to illustrate the methodology's outputs and quality.

5.5.4. Evaluation of the Architecture Generation Capability

This section evaluates the capability of the developed methodology to generate technically valid and internally consistent system architectures under new boundary conditions. The assessment follows the evaluation framework defined in Section 3.4.2, focusing on the technical execution quality of the LLM-based design process and the engineering validity of the resulting architectures. The goal is to determine whether the methodology enables the derivation of feasible, evidence-based ECS concepts that converge toward established engineering solutions such as the Boeing 787 electric ECS.

Technical-Systematic Evaluation

The technical-systematic evaluation examines the quality of the methodology's execution across three critical dimensions: prompt adherence and process execution, mathematical accuracy in engineering calculations, and demonstration of genuine technical understanding through independent problem-solving capabilities.

Prompt Execution Quality:
The assessment of prompt execution quality measures the extent to which the LLM followed explicit task instructions and methodological specifications across four sequential development phases. To ensure a systematic evaluation, prompt execution quality was decomposed into four dimensions: *instruction adherence*, *format consistency*, *completeness*, and *color-coding accuracy*. Each dimension was rated on a five-point scale, resulting in a quantitative profile as summarized in Table AT.29. Due to readability and the length of the full evaluation, detailed matrices are provided in Appendix A.13 on page 228.

The aggregated evaluation yields an overall rating of 4.75/5, demonstrating high reliability in task implementation. The LLM achieved perfect scores in instruction adherence (5/5), format consistency (5/5), and completeness (5/5), successfully creating all required deliverables, including the "Component Reusability Matrix", a complete requirements specification, and two fundamentally different physical architectures with comparative analysis.

[5]The visualization test was conducted in May 2025 using several state-of-the-art language models (GPT, Claude, Gemini), none of which were capable of generating the required technical schematics. Nevertheless, considering the fast-paced technological progress in this domain, such capabilities may emerge in future model versions.

Minor deviations were only observed in the *color-coding accuracy* dimension (4/5), where the transparency marking became partially inconsistent during later development phases as the model's focus shifted toward complex thermodynamic calculations.

Mathematical Accuracy and Engineering Feasibility:
The evaluation of mathematical accuracy proved particularly challenging and yielded significant methodological insights for LLM-based quantitative engineering applications (complete calculation verification in Appendix A.13 on page 230). While the final results demonstrate high accuracy (5/5), achieving reliable mathematical performance required iterative refinement of prompt strategies.

Early prompt versions with explicit formula specification led to systematic calculation errors despite correct parameter substitution. A characteristic example occurred in calculating HPC Stage 1 outlet temperature, where the LLM correctly applied the adiabatic compression formula

$$\begin{aligned} T_2 &= T_1 \cdot \left(\frac{p_2}{p_1}\right)^{\left(\frac{\kappa-1}{\kappa}\right)} \cdot \frac{1}{\eta} \\ &= 223\text{K} \cdot \left(\frac{0.50}{0.25}\right)^{0.286} \cdot \frac{1}{0.85} \\ &= 320\text{K} \end{aligned} \tag{5.1}$$

but incorrectly calculated 263 K (-10°C) instead of the correct 320 K (47°C). Remarkably, the subsequent Kelvin-Celsius conversion was correct, indicating that LLMs work more reliably with simple arithmetic operations than complex exponential functions.

Mathematical reliability was achieved only through explicit integration of JavaScript calculation scripts into the prompt structure. This methodological adaptation enabled consistent thermodynamic calculations, forming the basis for accurate architecture evaluation. The final system correctly calculated pressure budgets, power requirements, and thermal profiles across all components, with systematic verification procedures ensuring engineering plausibility.

Technical Understanding:
Assessment of technical competence achieved perfect scores (5/5 across all dimensions, Appendix A.13). The LLM demonstrated independent recognition of design flaws and systematic solution development. In Architecture 1, it identified the manifold pressure shortfall (0.76 vs. $\geq$0.80 bar) without hints, noting that an HPC ratio of $\sim$4.0 was insufficient once downstream losses of 0.20-0.24 bar were considered. The system then executed a complete engineering problem cycle: problem analysis, evaluation of three alternatives (higher HPC pressure, additional compression stage, or loss reduction), justified modification (single-stage HPC at $\sim$1.1 bar), impact analysis (temperature and power implications), and verification through recalculation.

The trade-off analysis further demonstrated engineering reasoning. For example, it compared power and temperature differences between the two architectures: "*Architecture 1 requires $\sim$400 kW total HPC power but yields higher outlet temperature (140°C). Architecture 2 consumes 570-600 kW but offers lower HPC stage temperatures, improving reliability and cooler life.*" Such comparisons show systematic evaluation of competing requirements and transparent documentation of design rationale.

Elegant Design Evaluation

The elegant design evaluation followed the framework by Watson and Griffin [179], which assesses the quality of engineering solutions along four dimensions: *effectiveness*, *robust-*

ness, *efficiency*, and *minimization of unintended consequences* (see Section 3.4.2). While the previous evaluations primarily examined methodological and computational accuracy, this assessment focuses on the quality of the generated system concepts themselves. A condensed summary of the main results is presented below; the full matrices and detailed reasoning are provided in Appendix A.13 on page 233.

Comparison with the Boeing 787 Reference:
A key result of the evaluation is the convergence of both generated architectures toward the design philosophy of the Boeing 787 electric ECS, despite the complete absence of this reference system from the knowledge base. Both concepts independently reproduced the fundamental principle of substituting engine bleed air with electrically driven compression, which illustrates the methodology's ability for principled reasoning and recombination.

Architecture 1 (Single-Stage Compression):
Architecture 1 achieves solid engineering performance with ratings of 4/5 for *effectiveness* and *efficiency*. The concept fulfills all major functional targets, maintaining a manifold pressure of at least 0.80 bar, a delivery temperature below 25 °C, and a mass flow of 1.8 kg/s while keeping power demand relatively low at around 400 kW. However, the single-stage high-pressure compressor leads to elevated outlet temperatures of approximately 140 °C, which reduces *robustness* (3/5) and may create integration challenges for thermal management and component reliability.

Architecture 2 (Two-Stage Compression with Intercooling):
Architecture 2 improves thermal robustness through intercooling between two compressor stages, resulting in lower outlet temperatures and a higher rating in *robustness* (4/5). This benefit, however, comes at the cost of increased system complexity and significantly higher power requirements (570-600 kW), which reduce *efficiency* (2/5). Both architectures fulfill all basic functional and performance requirements, but they illustrate the trade-offs between simplicity, energy demand, and operational stability that characterize early-stage conceptual designs.

Overall Quality Assessment:
When compared to the optimized Boeing 787 eECS reference, which achieves 5/5 in all four criteria, the LLM-generated architectures received overall ratings of 3.5/5 (Architecture 1) and 3.25/5 (Architecture 2). The B787 system demonstrates superior optimization through regenerative heat recovery, redundant subsystems, and seamless integration of thermal and electrical domains, improving overall aircraft efficiency by approximately 3 % through the elimination of bleed-air extraction. In contrast, the LLM designs represent technically consistent but preliminary solutions that capture the fundamental principles without achieving full integration or optimization.

Methodological Insights:
The results reveal a crucial distinction between process quality and solution quality. Although the methodology achieved near-perfect scores in technical execution and mathematical accuracy, optimal system solutions did not automatically follow. The generated architectures exhibit systematic reasoning and valid engineering logic but remain at the level of well-structured conceptual drafts. This finding confirms that LLM-based engineering processes can replicate the reasoning patterns of design synthesis but currently lack the depth of multi-objective optimization and cross-domain trade-offs achieved by experienced human engineers.

Conclusions:
The evaluation validates the methodology's capability to autonomously generate technically consistent architectures and converge toward proven engineering principles, while highlighting the continued necessity of an engineer-GenAI collaboration. SAGE accelerates early conceptual development and provides technically coherent starting points, but engineers' expertise remains indispensable for design refinement, optimization, and

validation. In this sense, the approach serves as a complementary tool that enhances engineering capability rather than replacing it, offering a practical pathway for integrating GenAI into the established practice of designing aircraft systems.

5.6. Summary

All in all, the ECS case study followed a clear sequence. After data preparation (Phase 1) and knowledge extraction (Phase 2), the initial focus was on whether system descriptions from literature could be systematically synthesized (Phase 3). This proved to be feasible. The generated document captured implicitly formulated information and provided a consistent, source-grounded representation of the conventional ECS.

Building on this result, it was investigated whether the methodology could perform a reuse/adapt/new analysis and then enable the step towards developing new concepts (Phase 4). The results were equivocal. While the reuse/adapt/new analysis and the subsequent RFLP documentation, including logical structures and component analysis, were convincing, the generation of complete architectures remained at the level of initial, technically consistent drafts.

Answering the research questions:

- RQ1 (Recursive Retrieval): The process for recursive retrieval from literature sources was tested empirically for the first time and was able to generate a comprehensive, comprehensible QA corpus. The functionality of the developed mechanism has thus been confirmed.
- RQ2 (Structured Synthesis): The proposed structured synthesis was also successfully validated: the fragmented QA-pairs resulted in a consistent, source-supported RFLP* document with high accuracy and complete traceability.
- RQ4 (Systematic Recombination): The methodology made it possible to generate new, bleed-free architectures from conventional ECS concepts, which independently converged on the basic principle of electrical compression. The results can be evaluated as technically consistent initial designs that point to innovation paths but do not yet achieve the optimization depth of industry-ready solutions.

Overall, the case study confirms that the methodology provides a solid foundation for systematic documentation and component analysis, while also demonstrating its potential to support new architectures. Final refinement and optimization, however, remain the task of engineers.

This methodology positions GenAI-generated outputs as informed starting points requiring engineering evaluation rather than final solutions, thereby addressing critical concerns about AI reliability in safety-critical contexts while leveraging AI capabilities to accelerate knowledge synthesis and processing tasks that currently consume the majority of engineering time. This perspective aligns with Garry Kasparov's reflection on human-AI collaboration in chess: "*Weak human + machine + better process was superior to a strong computer alone and, more remarkably, superior to a strong human + machine + inferior process*" [194]. The decisive factor is not the computational power of the GenAI, but the design of the collaborative process in which human expertise and AI capabilities are combined. Transferred to aerospace engineering, the strength of the SAGE methodology lies precisely in this principle: GenAI does not need to be the perfect engineer, it must be embedded into good engineering processes that allow human-GenAI teams to surpass what either could achieve independently. The intention is not to replace engineers but to create human-GenAI collaborative workflows.

6. Case Study 2: Thermal Management System

Following the evaluation in Case Study 1 (Chapter 5), this second case study applies the SAGE methodology to a new technical domain: the thermal management system (TMS) for aircraft fuel cell systems (FuCS). The objective is to examine the methodology's transferability beyond a known aircraft system. In the ECS case, the methodology was tested on its ability to derive architectures that converge toward the characteristics of an established reference system, the Boeing 787 eECS, based solely on conventional ECS documentation. In contrast, the present TMS study addresses a system type for which no comparable aircraft reference exists. It evaluates whether the same methodological sequence of data acquisition, recursive knowledge extraction, synthesis, and architecture generation can yield consistent and physically meaningful results.

This use case thus marks the first application of the methodology to a domain lacking a defined aircraft system, where the underlying data consists exclusively of heterogeneous and cross-domain information. It investigates whether SAGE can integrate insights from related sectors such as automotive and industrial fuel cell systems, extract transferable principles, and synthesize a coherent architecture tailored to the specific boundary conditions of aircraft operation. In doing so, the study complements the ECS case: together, the two form a methodological pair that demonstrates both reproducibility in a known domain and adaptability in an emerging one under near-greenfield constraints.

The following sections document the complete process of applying the methodology to the design of a TMS. The case leverages existing knowledge from adjacent technical domains while enabling the generation of new solutions adapted to the unique requirements of aerospace integration. The resulting design concepts are then evaluated. For detailed background on FuCS and TMS, readers may refer to Appendix A.14 (page 237).

6.1. Use-Case: Thermal Management System

Before defining the assumed system architecture for this case study, it is helpful to recall the broader architectural concepts, namely the More Electric Aircraft (MEA) and the potential role of FuCS. A central trend in modern aircraft system design is the progressive electrification of on-board systems. Under the MEA concept, hydraulic and pneumatic systems are increasingly replaced with electrical systems [195, p.7]. Different architectural concepts exist, ranging from partial electrification in current aircraft (e.g., Airbus 350, Boeing 787) to the long-term goal of an All-Electric Aircraft (AEA), where all secondary power is supplied electrically [196, 197]. Within this trend, a FuCS is being discussed as a potential electrical power source for large transport aircraft.

Of the different fuel cell types, Proton Exchange Membrane Fuel Cells (PEMFCs) are considered the most promising option for aerospace applications. Their advantages include structural simplicity, robustness against vibrations, and a high technology readiness level [198, p.6764]. Additionally, PEMFCs feature fast dynamic response and excellent start-stop behavior with reaction times below one second. They can operate safely under the

high mechanical loads typical for aircraft, such as varying ambient pressures, vibrations, and accelerations, as a result of their solid polymer electrolyte design. Unlike other fuel cell types, PEMFCs can function in any orientation, which is particularly advantageous for aircraft integration. Their operating temperature generally lies between 60-80°C, and they achieve electrical efficiencies of 50-68% per cell with high specific power densities of up to 1000 W/kg [199, p.9], [200, p.30f], [201, p.77], [202, p.2], [203]. They also offer reliable cold-start capabilities down to -20°C, which is essential for aircraft operation.

A PEMFC generates electricity through the electrochemical conversion of hydrogen and oxygen into water [199, p.2], [3, p.1]. The only by-products are heat and water. Despite its high efficiency, substantial amounts of heat are generated that must be dissipated to maintain optimal performance, ensure safety, and maximize the system's lifespan [199, p.19], [204], [205, p.14, 120]. As such, the development of dedicated TMS concepts for large-scale fuel cell applications in aircraft represents a key enabling technology for advancing sustainable aviation. In the following, the term *Fuel Cell System* (FuCS) refers exclusively to PEMFC technology.

6.1.1. System Architecture and Boundary Conditions

The development of a TMS for aircraft FuCS requires a thorough understanding of both the broader aircraft systems architecture and the specific constraints under which the TMS must operate. In this context, it is important to distinguish between two notions of architecture used in this chapter:

- Assumed architecture: A conceptual baseline of the surrounding aircraft and FuCS configuration that provides the operational context and boundary conditions for TMS design. It does not represent an object of development in this work but rather a defined set of assumptions that serve as input for the TMS design process.
- Generated architecture: The system concepts produced by the SAGE methodology that represent potential implementation pathways for the TMS itself.

The following subsection introduces the assumed aircraft and FuCS architecture, which defines the solution space for the TMS use case.

6.1.2. Assumed Aircraft Architecture

The aircraft architecture assumed for this dissertation is a potential configuration based on the MEA concept, wherein traditionally hydraulic and pneumatic systems are replaced with electrical counterparts [195]. As illustrated in Figure 6.1, the proposed electrified aircraft system architecture is powered by a FuCS[1] rather than conventional engine-driven generators. This would constitute an independent power supply for aircraft systems that is decoupled from the propulsion system.

Such an architecture, in which the electrical supply is provided entirely through the FuCS, might reduce the need to extract secondary power from the engines and could therefore offer several potential advantages:

- Enhanced high-pressure compressor efficiency: Less to no bleed air extraction from the high-pressure compressor stages is required for the ECS or wing anti-icing system [18, pp.7-8].

[1]The number of FuCS units is left unspecified here, as it depends on redundancy requirements driven by aircraft-level failure probability targets.

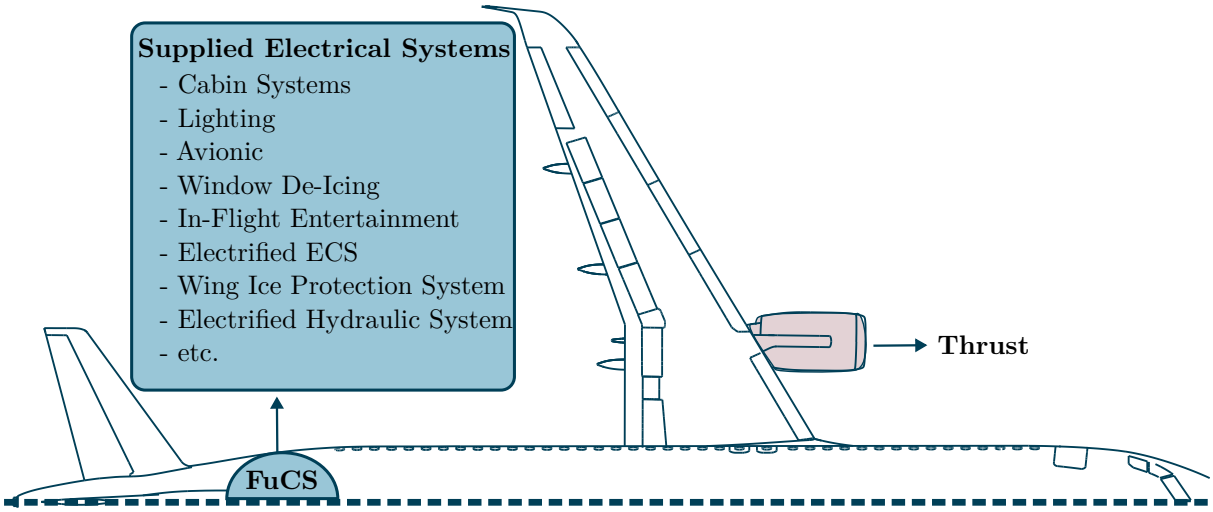

Figure 6.1.: Assumed power supply aircraft architecture with fuel cell power supply

- Improved turbine efficiency: As the electrical systems are powered by the FuCS and the hydraulic systems are electrified, no mechanical coupling between the accessory gearbox and the high-pressure shaft is necessary via angle gearboxes.
- Elimination of the accessory gearbox: Using an alternative power supply removes the gearbox requirement and leads to a reduction in system weight.
- Reduced maintenance costs and increased reliability: Removing pneumatic system components (precoolers, control valves, piping) simplifies engine design [18, p.8].

For this dissertation, it is assumed that the electrical systems would be continuously supplied with power from the FuCS throughout the entire flight and ground operation.

6.1.3. Assumed Fuel Cell System Architecture

The assumed FuCS architecture comprises multiple fuel cell stacks to provide the required power output. Each fuel cell stack consists of a series of individual fuel cells connected in sequence, combining their voltage or current to achieve the desired electrical power level. The architecture incorporates the following major subsystems:

1. Air supply subsystem: Compressors and air treatment components that supply compressed, filtered air to the cathode side of the fuel cell stacks.
2. Hydrogen supply subsystem: Regulates hydrogen delivery from storage tanks to the fuel cell anode side at appropriate pressure and flow.
3. Water management subsystem: Components for collecting, storing, and potentially reusing produced water from the electrochemical reaction.
4. Power electronics: Conversion and conditioning equipment, such as DC/DC converters and inverters, converts the direct current (DC) produced by the fuel cell stacks into the voltage and frequency levels required by the aircraft electrical systems.
5. Control and monitoring subsystem: Sensors, controllers, and software for operational management and safety monitoring.
6. Thermal management subsystem: The focus of this study is responsible for maintaining optimal temperature conditions across all components of the FuCS.

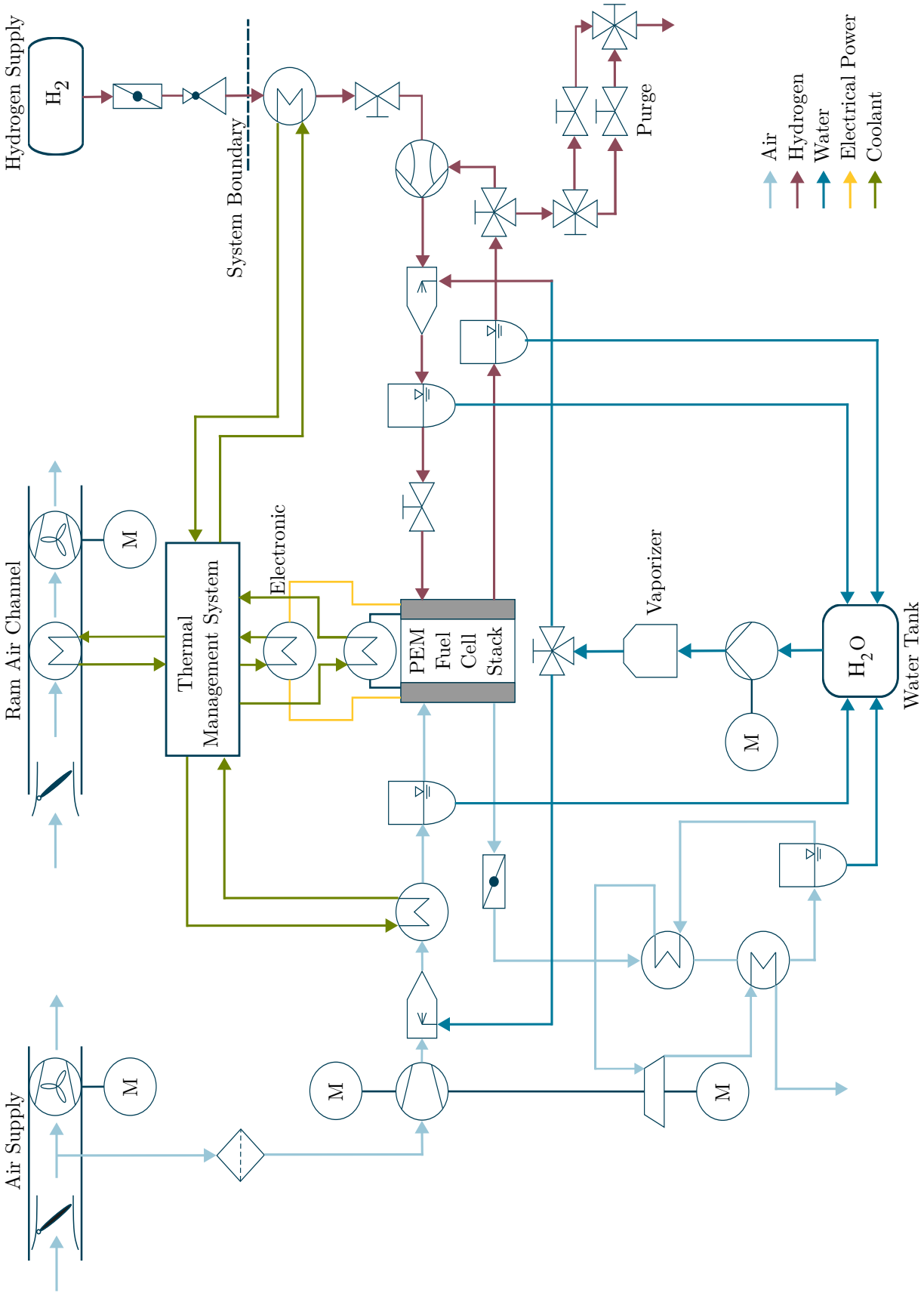

Figure 6.2.: Assumed FuCS system architecture, adapted from [206]

A schematic representation of the assumed FuCS architecture is shown in Figure 6.2, highlighting the thermal interfaces between subsystems. As can be seen, the thermal management subsystem interfaces with multiple heat sources and sinks within the overall system architecture. Primary heat sources include the fuel cell stacks, the air compressors and associated motors, and the DC/DC converters. Available heat sinks within the aircraft environment include the ram air (with temperature varying from +45°C at ground level to -56°C at cruise altitude) and the heat exchanger to heat the cryogenic hydrogen to a certain temperature level.

The calculation of precise boundary conditions for a fuel cell TMS requires detailed thermodynamic analyses under various operating conditions. Such calculations are not the focus of this dissertation. Instead, this work relies on a set of estimated boundary conditions. The corresponding assumptions, calculation steps, and formulae are documented in Appendix A.15. There, tables for heat quantities (Table AT.37), interface parameters (Table AT.38), and environmental conditions (Table AT.39) are provided as reference input for dimensioning.

6.2. Phase 1: Data Preprocessing

The development of an effective RAG process requires a comprehensive knowledge base encompassing relevant technical literature, standards, and patents. For the specific case of an aircraft fuel cell TMS, a systematic approach to data acquisition was implemented.

6.2.1. Scope of Processed Data

The data acquisition strategy employed a hybrid approach combining automated database searches with expert-guided manual curation to ensure optimal coverage of the specialized domain of aircraft fuel cell TMS, as described in Section 3.1.

A targeted search strategy was developed using hierarchical keyword combinations across three categorical dimensions:

1. Primary Keywords - Thermal management concepts:
   “Thermal Management System”, “TMS”, “Cooling System”, “Cooling Circuit”, “Heat Management”, “Temperature Control”, “Thermal Control”, “Thermal Design”
2. Secondary Keywords - Fuel cell specifics:
   “Fuel Cell”, “PEMFC”, “Hydrogen Fuel Cell”, “Fuel Cell Stack”, “FuCS”
3. Tertiary Keywords - Components and context:
   “Heat Exchanger”, “Radiator”, “Coolant”, “Pump”, “Architecture”, “Design”

These keyword sets were programmatically combined into structured queries and used for automated database searches (see Section 3.1.1). In addition, targeted direct queries (e.g., “Fuel Cell TMS”, “PEMFC Cooling System Design”, “Aircraft Fuel Cell Thermal Management”) were incorporated to capture highly relevant documents that might otherwise be missed. The search was conducted across three primary scientific databases: Scopus, Springer Nature, and arXiv, with systematic filtering to extract the most relevant contributions in this field.

To complement automated searches, manual identification of patent literature and standards was conducted, focusing on thermal management approaches and regulatory requirements specifically applicable to aircraft fuel cell systems. As API access to patent

databases and standards organizations was unavailable, relevant documents were identified through manual searches instead of automated retrieval.

Through this approach, a focused knowledge base of 115 documents was compiled:

- Aircraft Standards (3 documents): SAE ARP4754B and ARP4761A for safety-critical system design, CS-25 regulations for commercial aircraft certification
- TMS Patents with FuCS Focus (16 documents): Patents documenting novel approaches to fuel cell cooling systems, new heat exchanger (HX) designs, and integrated thermal architectures specifically developed for mobile and aircraft applications
- Fuel Cell Specific TMS Scientific Papers (58 documents): Specialized literature addressing unique challenges of fuel cell thermal management, including stack temperature control, waste heat recovery, and integration with aircraft systems
- TMS General Applications (38 documents): Comprehensive thermal management principles applicable across different domains, providing foundational knowledge for system architecture development, including heat exchanger design, coolant selection, and thermal control strategies

Although 115 documents provide strong coverage (standards, patents, and domain literature), no directly comparable TMS architecture at the required power and integration level was found. This resulted in a trade-off between broad thematic coverage and limited architectural specificity. To mitigate bias, sources from adjacent domains (automotive, stationary, marine) were included and later constrained by aircraft boundary conditions (Appendix A.15, page 248). This design choice intentionally stresses SAGE's ability to generalize under heterogeneous evidence.

6.3. Phase 2: Extended RAG

Phase 2 builds on the curated document corpus established in Phase 1 and applies the extended retrieval-augmented generation process. The purpose of this phase is to transform the literature base into structured knowledge that can be used for subsequent synthesis and architecture generation. The following subsections present the high-level questions used to initiate the exploration of the TMS domain, followed by a statistical evaluation of the QA pairs across categories and hierarchy levels.

6.3.1. High-Level Questions for Literature Exploration

To systematically explore the knowledge domain of aircraft fuel cell TMS, a structured catalog of domain-specific questions was developed, following the methodology established in Section 3.1.4 and applied in the ECS case study (Section 5.3.1). The catalog encompasses the six defined system aspects (RFLP*) and has been adapted to address the specific challenges of TMS in aircraft FuCS.

These TMS-specific guiding questions serve as an entry point for the recursive knowledge exploration process, enabling systematic investigation of thermal management challenges unique to aircraft fuel cell applications. The complete catalog of 30 guiding questions across all six categories is documented in Appendix A.16 (page 258), providing the foundation for the knowledge synthesis and architecture generation. The systematic exploration through these questions enables identification of thermal management solution patterns,

critical design parameters, and integration requirements that form the basis for developing new TMS architectures tailored to aircraft fuel cell systems.

6.3.2. Statistical Evaluation of Question-Answer Generation

Following the recursive knowledge acquisition process described in Section 3.2 and applying the same methodology as established in Section 5.3.2, 930 QA-pairs were generated across five hierarchy levels over the extended RAG. The 115 TMS-specific documents enabled comprehensive coverage, with 867 questions answered (93.2%). Figure 6.3 illustrates the overall coverage trends across all categories and hierarchy levels. As shown, the answer rates remain consistently high, confirming that the corpus offers a dense knowledge base. The detailed numerical distributions are summarized in Tables 6.1 and 6.2.

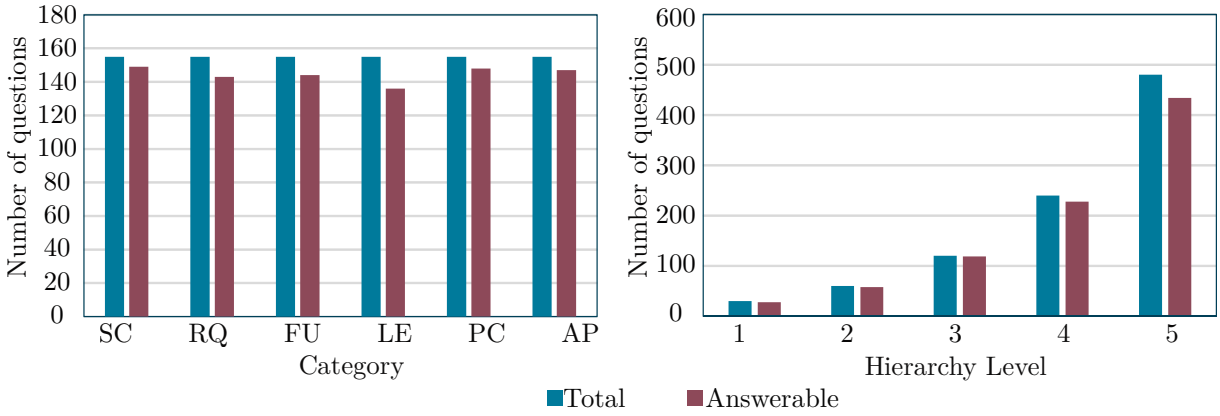

Figure 6.3.: Answered TMS questions by category (left) and hierarchy level (right). Blue bars show all analyzed questions, while red bars denote the subset that could be answered with the information available in the database (Answerable).

Table 6.1 shows the distribution of answered questions by category, with all six system aspects exceeding a 93% answer rate. Questions regarding the system context achieved the highest answer rate at 96.1%, demonstrating excellent capture of operational boundaries, environmental constraints, and stakeholder requirements (see Table 6.1). Questions, with a focus on functions, reached 92.9%, indicating comprehensive coverage of thermal management capabilities and operational behaviors. Physical component questions

Table 6.1.: Overview of answered questions by category

Category	Total Questions	Answered Questions	Answer Rate
System Context (SC)	155	149	96.1%
Requirements (RQ)	155	143	92.3%
Functions (FU)	155	144	92.9%
Logical Elements (LE)	155	136	87.7%
Physical Components (PC)	155	148	95.5%
Architecture Patterns (AP)	155	147	94.8%
Total	**930**	**867**	**93.2%**

achieved 95.5%, showing strong documentation of hardware implementations, materials, and component specifications. Questions in terms of architecture patterns and requirements achieved 94.8% and 92.3%, reflecting excellent coverage of design patterns and system specifications, with the curated corpus effectively capturing both abstract design knowledge and concrete requirements.

The coverage by hierarchy level is summarized in Table 6.2. Even at deeper levels of recursive questioning, answer rates remain high, with 99.2% at Level 3 and 95% at Level 4. At the most detailed Level 5, 90.4% of questions could still be answered, confirming that the corpus contains sufficient technical depth for architectural reasoning.

Table 6.2.: Overview of answered questions by hierarchy level

Hierarchy Level	Total Questions	Answered Questions	Answer Rate
Level 1 (Initial)	30	28	93.3%
Level 2	60	58	96.7%
Level 3	120	119	99.2%
Level 4	240	228	95%
Level 5	480	434	90.4%
Total	**930**	**867**	**93.2%**

To illustrate the level of detail captured, the following QA-pair from the requirements category is provided as an example in Box 11. This example illustrates several important aspects of the recursive knowledge acquisition process. First, it shows that the generated questions can target highly specific coordination mechanisms between components, going beyond generic system descriptions. Second, the corresponding answer demonstrates that the curated corpus provides sufficient technical depth to capture control strategies, response times, and trade-offs between accuracy and efficiency. Such detail highlights the capability of the extended RAG process to extract actionable engineering knowledge rather than only high-level summaries.

Compared to the ECS baseline (165 answered questions from 11 documents, 17.6% answer rate), the TMS corpus yielded 867 answered questions from 115 documents (93.2% answer rate). This nearly fivefold increase in answered questions and improvement in answer rate underscores the decisive role of targeted corpus curation and sufficient document coverage for effective knowledge extraction. In summary, the statistical analysis confirms that the curated TMS corpus provides both breadth and depth of information, forming a robust foundation for subsequent knowledge synthesis and architecture generation.

The key question for the subsequent synthesis phase becomes: Can the LLM leverage these 867 answers, representing comprehensive knowledge across all six categories and five hierarchy levels, to generate thermal management architectures and detailed system documentation for aircraft fuel cell applications? This question directly addresses the practical applicability of the methodology in specialized engineering domains, where focused, high-quality documentation can form the basis for new architectural concepts.

6.4. Phase 3: LLM Knowledge Synthesis

The knowledge synthesis phase marks the transition from isolated QA-pairs to a coherent and interpretable body of engineering knowledge. While the preceding ECS case study

Box 11: Example of an QA-pair at level 3

LLM-generated question:
What are the specific interactions and coordination mechanisms between the coolant pump, fan, and bypass valve controls to ensure stable temperature uniformity while minimizing parasitic power losses during rapid load changes?

Answer (without thought process):
The coolant pump, fan, and bypass valve controls interact through a coordinated temperature-based feedback control system to maintain stable temperature uniformity while minimizing parasitic power losses during rapid load changes. Specifically:

- The **coolant pump** modulates the coolant mass flow rate based on the temperature difference between the coolant inlet and outlet of the fuel cell stack, adjusting heat removal capacity dynamically.

- The **bypass valve** regulates the coolant flow path by mixing coolant bypassing the radiator with coolant passing through it, controlling the coolant temperature entering the pump and stack. This mixing ensures temperature uniformity and prevents thermal shocks.

- The **fan** controls the airflow through the radiator, adjusting heat dissipation to ambient air based on the radiator coolant outlet temperature.

These components are controlled via PI controllers with anti-windup schemes or advanced controllers like LADRC or MPC to compensate for system inertia and delays (e.g., fan response time ~ 8s). The system is decoupled into coolant flow and air flow control loops, each optimized to respond to temperature changes and disturbances rapidly but smoothly. During rapid load changes, the control system anticipates thermal disturbances and adjusts pump speed, fan speed, and bypass valve position in a coordinated manner to maintain the fuel cell stack temperature within the optimal range (60--80° C), ensuring uniform temperature distribution and minimizing parasitic power consumption by avoiding excessive actuator activity.
Trade-offs exist between temperature control accuracy and energy efficiency; more precise control strategies consume more parasitic power but yield better temperature stability, while simpler strategies reduce power consumption at the expense of larger temperature fluctuations.

had already demonstrated the methodology's capability to generate quantitatively assessable and traceable synthesis reports, the focus in the TMS case was placed on the qualitative assessment of synthesis quality and engineering correctness. In contrast to the ECS, which benefits from decades of consistent documentation and standardized terminology, the TMS for FuCS integration is characterized by a highly fragmented knowledge base. Available sources include academic studies, patents, and conceptual design papers that differ substantially in scope, level of abstraction, and use of technical terms. Accordingly, this phase primarily examined whether a coherent and technically plausible synthesis report could be generated at all under these heterogeneous boundary conditions.

The central question, therefore, was not quantitative performance but the model's capability to consolidate dispersed information into an engineering-consistent representation of the TMS. Specifically, the synthesis sought to verify whether the LLM could (i) represent fundamental heat-management principles consistently, (ii) maintain technical correctness across system interfaces and energy flows, and (iii) achieve sufficient completeness to support the subsequent architecture-generation phase.

6.4.1. Prompt Engineering for Knowledge Synthesis

For the TMS case, prompt engineering required a re-conceptualization compared to the ECS case. While the ECS prompts were oriented toward documenting and varying a well-defined system, the TMS prompts had to consolidate heterogeneous knowledge from multiple technical domains and condense it into coherent architectural building blocks. This methodological adaptation was necessary because no unified reference system or consistent system context exists for the targeted power class of fuel-cell-based TMS. The literature spans numerous application areas such as automotive, marine, stationary, and laboratory systems, each with distinct boundary conditions, operating environments, and performance requirements. Consequently, the system context module that had formed an integral part of the ECS synthesis was deliberately omitted. Instead, a generalized and technology-agnostic prompting strategy was adopted to derive transferable design principles that remain valid across these diverse operational domains.

The TMS knowledge synthesis process employs four specialized prompts that ensure a systematic progression from universal requirements to concrete implementation options. The different prompt modules are the following:

- Module 1: Requirements Analysis:
  The first prompt extracted universal requirements for thermal management systems from the complete literature corpus. In contrast to the ECS case, where system-specific requirements were documented, the TMS synthesis derived cross-domain and technology-independent constraints, such as temperature regulation, reliability, redundancy, and integration with fuel-cell stack operation.

- Module 2: Functional Analysis
  The second module performs a comprehensive function extraction, identifying and categorizing all TMS functions mentioned in the literature. In contrast to the ECS case, which described functions of an established system, this case study extracts all generally applicable functions from the cross-domain literature. It evaluates their applicability to various fuel cell configurations.

- Module 3: Logical Elements Analysis
  The logical element analysis focuses on abstract system elements that fulfill specific functions within the TMS, independent of their physical implementation.

- Module 4: Components Analysis
  The final module creates an exhaustive component inventory of all physical TMS elements mentioned in the literature. This comprehensive component analysis forms the basis for subsequent architecture development through systematic identification of available technologies and their applicability to new system configurations.

Each module followed a standardized prompt structure to ensure methodological consistency and comparability of results. The general TMS prompt comprised the following elements:

- Task Definition: "*Extract and synthesize universal thermal management system {X} for PEM fuel cell applications from the literature database, with focus on technology-agnostic, application-universal principles.*" Here, {X} denotes the respective analysis domain (requirements, functions, logical elements, or components). The prompt explicitly defined the target domain, the analytical scope, and, where applicable, its dependency on the results of preceding modules.

- Output Format Specification: All results were required in structured HTML format. Tables and lists were mandated instead of diagrams to ensure consistent formatting, readability, and comparability across all modules.

- Document Structure Definition: Each prompt predefined a standardized document layout aligned with the RFLP logic, typically including an executive summary, tables, traceability tables, and detailed specifications
- Quality and Traceability Goals: Each output had to be comprehensive, internally consistent, verifiable, and fully traceable to source QA-pairs or derived module IDs. Missing information was to be explicitly marked as assumptions (red) with stated confidence level and verification needs.
- Cross-Domain Adaptation: Because the TMS synthesis integrated knowledge from heterogeneous domains (automotive, marine, stationary, laboratory), the prompt explicitly required a comparative presentation of results across these fields. Applicability matrices, consistency ratings, and requirement universality analyses were therefore embedded as default sections in each module. This ensured that the resulting knowledge representation remained technology-agnostic and transferable across future aircraft fuel-cell TMS designs.

As in the ECS case, a confidence classification system was used. Information directly traceable to QA-pairs and consistent across domains was marked green, results derived were marked orange, and extrapolations were marked red. This transparent marking ensured clarity about the reliability of the synthesized knowledge and guided subsequent evaluation steps.

6.4.2. LLM Configuration and Settings

For the TMS knowledge synthesis, the configuration followed the empirically evaluated setup established in Study E (Section 4.2.5). The model GPT-4.1 was employed, as it had demonstrated the highest performance and synthesis quality among all evaluated candidates. Maintaining methodological consistency with the ECS case ensured that potential differences in synthesis quality could be attributed to the characteristics of the underlying knowledge base rather than to model behavior.

The temperature parameter was again set to 0.2, supporting factual precision in the extraction and consolidation of the extensive literature data while preserving sufficient flexibility for the integration of heterogeneous cross-domain knowledge. All other parameters, including chunk size, retrieval depth, and recursion threshold, were aligned with the ECS configuration to guarantee comparability. This consistent setup ensured that the TMS synthesis primarily reflected the model's ability to maintain internal consistency, engineering correctness, and completeness under more heterogeneous boundary conditions.

6.4.3. Result of the Knowledge Synthesis

The knowledge synthesis produced a structured catalog of requirements, functions, logical elements, and components derived from the curated corpus. In contrast to the ECS case, which documented and varied a known system architecture, the TMS synthesis consolidated cross-domain design principles into a coherent baseline that serves as input for the subsequent architecture generation. This marks a critical methodological extension of the synthesis framework, as it demonstrates the applicability of the approach to a domain without a unified reference system. The results, therefore, represent a generalized, technology-agnostic foundation for TMS applicable across multiple fuel-cell domains. Selected excerpts illustrating the report structure and level of detail are shown below, with additional examples provided in Appendix A.17 on page 260.

A set of six core requirements was identified over the LLM, representing the essential minimum for any TMS configuration, see Table 6.3. This reduction to fundamental principles, ranging from temperature regulation of critical components through heat extraction and transfer to continuous monitoring of temperature, pressure, and flow parameters, highlights the synthesis focus on universally valid thermal-management requirements. Each of these requirements is indispensable for maintaining system functionality, thereby ensuring complete coverage of essential operational objectives.

Table 6.3.: TMS synthesis report - functional requirements hierarchy analysis

Requirement ID	Description	Critic. Level	QA-pair ID
REQ-FUNC-001	Regulate temperature of critical components (fuel cells, batteries, motors) to maintain safe and efficient operation	High	[FU.1]
REQ-FUNC-002	Extract and transfer heat from components to heat sinks or environment via heat exchangers and radiators	High	[FU.1]
REQ-FUNC-003	Maintain coolant quality by removing particles and deionizing to prevent clogging and electrical shorts	High	[FU.1]
REQ-FUNC-004	Coordinate thermal management across multiple subsystems to improve compactness, energy efficiency, and temperature control	High	[FU.1]
REQ-FUNC-005	Provide controlled heating to components during cold conditions to maintain operational temperature limits	Medium	[FU.1]
REQ-FUNC-006	Continuously measure temperature, pressure, and flow parameters and employ control strategies to maintain thermal stability and uniformity	High	[FU.1]

Building on this foundation, the LLM identified twelve distinct functions covering the TMS operating spectrum. From temperature control and heat transport to coolant quality management, health monitoring, and fault diagnosis, all critical capabilities are captured. An excerpt is shown in Table 6.4. A complete list (Table AT.41) is on page 261.

A methodological limitation became visible in the mapping between requirements and functions. Due to the sequential synthesis, some functions were extracted without directly corresponding to the requirements. Since the synthesis process first extracts the (functional) requirements and subsequently the functions, without enabling iterative feedback, certain functions were extracted for which no explicit functional requirement exists. This inconsistency results from the rigid sequential prompt structure and highlights the need for iterative refinement between synthesis levels. In practical system development, such inconsistencies would be resolved through dedicated feedback loops for requirement validation and functional decomposition.

Based on the identified functions, the logical element analysis established abstract elements that implement these functions and define the TMS interaction topology. Eight key logical elements were derived (Table 6.5), from heat exchangers and coolant pumps to control valves and thermal control units. A notable discrepancy is the absence of explicit filter elements despite the presence of the function "*Maintain Coolant Quality*" with the specific description "*Remove particles and deionize coolant to prevent clogging and electrical shorts*".

The physical components analysis, however, shows consistency with the logical elements and confirms the systematic coherence of the lower abstraction levels. The eight documented core components, from heat exchangers through coolant pumps to temperature sensors, completely map the physical implementation spectrum of the identified logical architecture.

Table 6.4.: Excerpt of the TMS function inventory - primary functions

Func. ID	Function Name	Function Description	QA-pair ID	Critic. Level	Related Components	Cross-Domain Applic.
FUNC-001	Control Temperature	Maintain the temperature of aircraft systems and cabin environment within acceptable operational and comfort limits	[FU.1]	High	Heat Exchang., Pumps, Sensors, Control Units	Universal
FUNC-002	Acquire Heat	Extract heat from diverse sources including engines, avionics, electrical equipment, hydraulic systems, and environmental control system components	[SC.1]	High	Heat Sources, Heat Exchang.	Universal
FUNC-003	Transport Thermal Energy	Transfer thermal energy via fluids, heat exchangers, ducts, pumps, and control equipment to heat sinks	[SC.1]	High	Heat Exchang., Pumps, Ducts, Valves	Universal
FUNC-004	Reject Heat	Safely dissipate heat to terminal heat sinks such as ambient air or fuel	[SC.1]	High	Heat Sinks, Heat Exchang., Fans	Universal
FUNC-005	Maintain Coolant Quality	Remove particles and deionize coolant to prevent clogging and electrical shorts	[FU.1]	High	Filters, Deionization Units	Universal
etc.						

Table 6.5.: Excerpt of the TMS logical elements registry

Elem. ID	Element Name	Element Description	QA-pair ID	Abstrac. Level	Interface Requirements	Cross-Domain Applic.
LE-001	Heat Exchanger	Device that transfers heat between fluids without mixing, designed to be lightweight, compact, and efficient, often with careful geometries	[PC.1]	Comp.	Fluid interfaces, therm. interfaces, pressure ratings	Universal
LE-002	Coolant Pump	Mechanical or electrical device that circulates coolant through thermal loops, sized to meet flow and pressure requirements	[PC.4]	Comp.	Elec. & mech. interfaces, flow rate and pressure control	Primarily aerospace
LE-003	Control Valve	Actuated device regulating fluid flow paths and rates within thermal loops, enabling dynamic control of coolant routing and heat rejection	[PC.4]	Comp.	Electrical actuation signals, flow control interfaces	Universal
etc.						

Overall, the synthesis achieved high internal consistency and engineering plausibility, even under heterogeneous source conditions. Minor omissions, such as missing filter components, reflect the limits of the universal prompt approach rather than conceptual inconsistencies in the synthesis itself. These findings underline the model's ability to generalize domain knowledge into a technically coherent and complete system representation, which serves as a robust basis for the subsequent architecture-generation phase.

6.4.4. Qualitative Evaluation of Knowledge Synthesis

The qualitative evaluation of the TMS knowledge synthesis examined the overall coherence of the results, the technical plausibility of the synthesized information, and the completeness of the captured system description. The synthesis exhibited a high level of structural consistency across requirements, functions, logical elements, and physical components. Functional dependencies were maintained without major contradictions, indicating that the LLM successfully captured the internal logic of the TMS domain despite the heterogeneous document base.

Particularly noteworthy is the model's ability to identify, differentiate, and hierarchically organize requirements, functions, and components with high semantic precision. Even in the absence of standardized templates, the structured prompt design enabled the model to reconstruct a consistent RFLP-like framework that mirrors the reasoning typically applied by engineers during early system design. This demonstrates that targeted prompt engineering and recursive synthesis can translate fragmented textual information into a coherent knowledge representation.

From a technical perspective, the synthesis achieved notable precision in extracting information from the database, such as the maximum permissible stack temperature difference ("*Max Stack* $\Delta T < 10°C$, *ideally* $< 5°C$") and allowable coolant conductivity ("$<$ *3* $\mu S/cm$"). Occasional contextual gaps, such as unspecified references for temperature ranges ("*60-80°C*"), highlight the limitations of domain attribution across heterogeneous sources. These inconsistencies underline the importance of engineering validation, particularly when quantitative data are transferred between domains with differing operational contexts. Variations in specification quality stem from uneven information density within the literature: some sources provide detailed parameters and traceable contexts, whereas others remain fragmentary.

Despite these limitations, the report represents a valuable starting point for developing new systems that do not yet exist in the desired performance class or specific application area. Unlike the ECS case, where an established system served as the basis, the TMS synthesis had to consolidate fragmented and cross-domain knowledge. The available documents describe TMS applications in the automotive sector, marine industry, or other industrial domains. Although individual TMS publications from the aircraft sector exist, these focus on significantly lower performance classes and therefore offer no direct reference to the targeted high-performance aircraft application. This challenge makes the successful knowledge synthesis all the more remarkable, as it provides a systematic overview of fundamental functions, basic requirements, and proven components despite the heterogeneous and non-specific literature base.

In summary, the TMS knowledge synthesis report offers a meaningful foundation for aircraft-specific TMS design. Its systematic structuring, quantitative precision, and transparent handling of uncertainty enable the development of coherent architectures, even though identified inconsistencies and methodological limitations highlight areas for further refinement.

6.5. Phase 4: LLM Architecture Generation

Following the evaluation of knowledge synthesis, the development of a TMS architecture is subsequently undertaken. In contrast to the ECS case, which could leverage well-documented reference architectures, the TMS must be derived without an established aircraft baseline. This phase, therefore, applies first-principle constraints and embedded calculation routines to transform cross-domain principles into a physically coherent aircraft architecture.

6.5.1. Prompt Engineering for Architecture Generation

In contrast to knowledge synthesis, which focused on extracting and structuring information, architecture generation required prompts oriented toward the creative application of this knowledge to solve specific design problems. The process was structured in two stages: first, a Requirements-Functions-Logical Elements (RFL) analysis as a conceptual bridge between abstract knowledge and aircraft-specific requirements, followed by the development of a concrete physical TMS configuration.

The RFL analysis prompt systematically projects the synthesized insights onto the specific requirements of the aircraft fuel cell application. The RFL prompt explicitly integrates the thermodynamic boundary conditions of various flight phases with the extracted TMS knowledge synthesis report into a coherent system specification. The second prompt focuses on the detailed physical implementation of a specific TMS configuration. To ensure systematic and comparable evaluation of the methodology, the system configuration was limited to a single-stage cooling system architecture with 50/50 water-glycol coolant. This restriction serves to control the assessment of architecture development capabilities, as it reduces complexity variability and enables a focused analysis of the design.

To guarantee physically consistent results, a JavaScript-based calculation environment was embedded directly into the prompts. This allowed the LLM to access predefined thermodynamic calculation functions, such as mass flow, coolant temperature, and pressure-drop calculations, rather than relying on heuristic reasoning. This ensured that pressure drops, heat transfer rates, and pump performance were dimensioned within feasible ranges. The excerpt Box 12 illustrates how coolant properties were programmatically defined using JavaScript within the prompt.

Box 12: Example of an used JavaScript within the prompt

```
**Use these specific JavaScript functions for ALL calculations:** 'JavaScript'
1. Coolant Property Functions
function getCoolantProperties(temperature_celsius) {
    50/50 Water-Glycol mixture properties
    const T = temperature_celsius;
    const density = 1020 - 0.5 * T; kg/m^3, temperature dependent
    const specificHeat = 3500 + 2 * T;  J/kgK, slight temperature dependence
    const viscosity = 0.0018 * Math.exp(-0.01 * T);  Pa*s, exponential
                    temperature dependence
    const thermalConductivity = 0.45 + 0.001 * T; W/mK

    console.log(Coolant properties at T° C:‘);
    console.log(‘Density: {density.toFixed(1)} kg/m^3, Cp:
     {specificHeat.toFixed(0)} J/kgK‘);

    return {density, specificHeat, viscosity, thermalConductivity};}
```

Each prompt in the architecture generation phase followed a standardized structure to ensure methodological transparency and reproducibility. As in the knowledge synthesis phase, this uniform prompt framework defined the essential building blocks of every model instruction, specifying objectives, output structure, transparency rules, and validation criteria. The architecture prompts comprised the following elements:

- Task Definition: The prompt explicitly specified the analytical scope, required traceability chains, and quantitative validation rules. It also defined the boundaries and constraints that all reasoning and calculations had to comply with. This included thermodynamic constraints (e.g., maximum stack inlet $< 70°C$, hydrogen conditioning inlet $> 60°C$), allowable pressure drops per component and total system (e.g., < 0.15 bar piping, < 0.4 bar HX), the use of a water/glycol coolant, flow-rate limits, redundancy requirements, and flight-phase dependent heat loads.
- Output Format Specification: All results were required in structured HTML
- Document Structure Definition: Each prompt predefined a uniform report layout aligned with the RFL* logic
- Analytical and Mathematical Integration: Both the RFL and architecture prompts incorporated embedded JavaScript functions for the calculation of coolant properties, heat loads, mass flows, and pressure drops. In addition to numerical consistency, these functions enforced compliance with the predefined boundary conditions and generated automated warnings whenever results exceeded realistic engineering limits (e.g., excessive Δp, unrealistic temperatures, or flow velocities). Validation routines such as `validateRealisticPressureDrops()` and component-level constraint checks ensured that every configuration adhered to thermodynamic, hydraulic, and operational feasibility limits.
- Color-Coding and Transparency System: A four-level color legend was used to document evidence origin and domain adaptation: green for literature-traceable values, orange for domain-adapted derivations, blue for water-glycol system specifics, and red for assumptions in areas without aircraft references. This explicit marking allowed subsequent validation of assumptions and design-critical parameters. This scheme ensured immediate visual traceability of data reliability and facilitated post-synthesis validation.
- Quality, Traceability, and Constraint Compliance: Each report had to maintain full traceability across abstraction levels: every requirement mapped to at least one function, each function was realized by one or more logical elements, and every logical element corresponded to a physical component. Quantitative constraint validation through embedded calculations was mandatory, and any violation of thermodynamic or geometric limits triggered iterative re-evaluation of the affected subsystem. Before completion, every generated architecture was verified against the core constraint set to confirm that all aircraft thermal loads were addressed, all heat exchangers operated within allowable temperature margins, and all pressure drops and flow rates satisfied engineering feasibility criteria.

6.5.2. LLM Configuration and Settings

For TMS architecture development, the GPT-o1 configuration, which was already successfully employed in the ECS evaluation, was retained. This continuity minimizes methodological deviations between the two application cases, thereby improving the comparability of the architecture development results.

The GPT-o1 model proved particularly suitable for TMS development, as it combines systematic problem decomposition with creative solution finding. The fundamental challenge lies in combining creative system synthesis with rigorous thermodynamic validation, a requirement that the GPT-o1 model automatically balances through its inherent chain-of-thought reasoning.

6.5.3. Architecture Generation

The architecture development for the TMS followed the established two-stage consolidation process, where individual HTML reports were systematically integrated into a technical document. The resulting document represents a complete TMS specification for fuel cell systems. In contrast to the ECS case, the Reuse/Adapt/New classification scheme was not applied in this phase. As the TMS represents a new system architecture that has not yet been realized in comparable aircraft applications, most design elements were developed as new concepts based on the defined boundary conditions rather than derived from existing systems.

Table 6.6.: Excerpt of logical elements mapped to physical TMS components

Log. Elem. ID	Logical Element Name	Physical Component	Component Specifications	Selection Rationale
LE-001	HX	Plate-Fin HX (FC Stack)	Custom geometry, A/C-grade aluminum, 0.2-0.3 bar drop	Very high effectiveness, compact design for aircraft space
LE-001	HX	Plate-Fin HX (DC/DC)	Scaled-down geometry, 0.1 - 0.2 bar drop	Similar design approach, smaller capacity & footprint
LE-001	HX	Plate-Fin HX (Air Cond)	Custom geometry for 30-35 kW capacity, 0.2 bar drop	Operates at high external temps (up to 129°C), requires robust core
LE-001	HX	Plate-Fin HX (H2 Cond)	High temperature gradient handling, 0.2 bar drop	For hydrogen warming from -252°C up to 60°C, needs careful design
LE-001	HX	Plate-Fin HX (Ram Air)	High capacity for ≈433 kW rejection, 0.3-0.4 bar drop	Main system heat sink, must handle large load quickly
LE-002	Coolant Pump	Electric Centrifugal Pump	Variable speed (50-200 LPM), 2.5-3.5 bar boost	Precise flow control, lightweight aerospace motor
LE-003	Control Valve	Solenoid/Flow Control Valve	0.05-0.1 bar drop fully open, fast actuation	Referenced design from commercial aerospace applications
LE-005	Thermal Control Unit	Digital TMS Controller	MPC/SMC fault-tolerant control loops, multiple sensor inputs	From literature on advanced TMS control in aircraft environment
LE-007	Load Shedding	Integrated TMS-FCU Logic	Software-based load priority algorithms	Monitors TMS capacity, aircraft power demands
LE-008	Alerting System	Flight Deck Alert Integration	Central alert prioritization, standard cockpit EICAS interface	Follows standard flight deck design guidelines
LE-005	Thermal Control Unit	Expansion/ Deion. Tank & Filter Module	∼5-10 l capacity, integrated deion. resin, -0.1-0.2 bar drop	Specifically designed for water-glycol mixture & aircraft altitude changes

A central outcome of this phase is the consistent implementation of RFL traceability. Table 6.6 provides an excerpt of the component specifications, showing how heat exchangers, pumps, valves, and control units were detailed with operating parameters suitable for aerospace applications.

Building on this mapping, the consolidated system architecture was derived. The resulting TMS is based on a 50/50 water-glycol coolant configuration, which had been defined as a boundary condition in advance. As illustrated in Figure 6.4, the coolant circuit is divided into two main branches followed by a final heat rejection stage. Branch A combines the fuel cell stack heat exchanger (440 kW) with the DC/DC converter heat exchanger (25 kW), requiring a total mass flow of about 12.6 kg/s under the boundary conditions defined in Table AT.38. Branch B couples the air conditioning system (36.3 kW) with the hydrogen conditioning heat exchanger (-68.1 kW), resulting in a combined flow of roughly 5.5 kg/s. In the final stage, the ram air heat exchanger rejects approximately 433 kW to the environment, thereby providing an overall heat balance for the system.

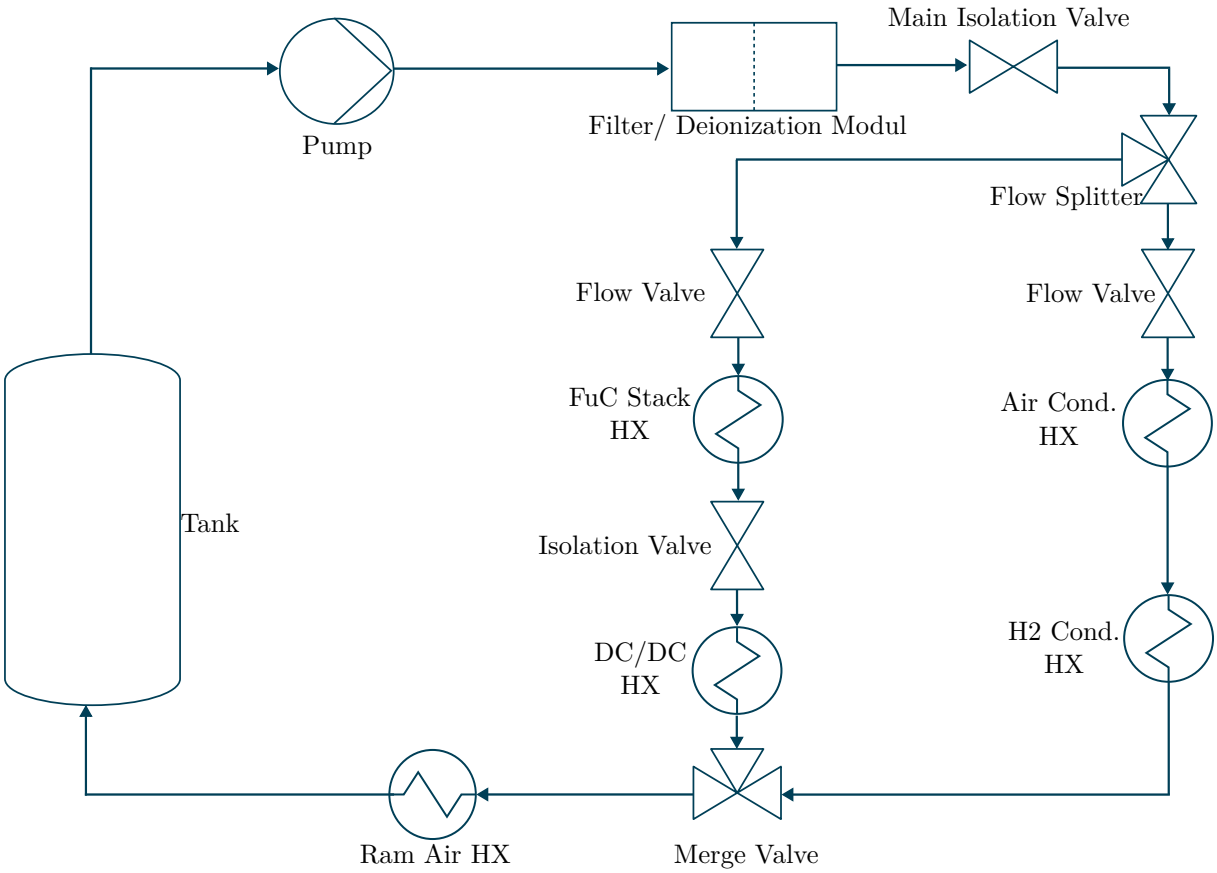

Figure 6.4.: Developed TMS architecture (reconstructed by the author from LLM-generated tabular specifications in Table 6.7).

The system architecture illustrated in Figure 6.4 was created manually by the author based on the validated data presented in Table 6.7. While the LLM successfully generated the complete tabular specification of components, mass flows, and boundary conditions, it was not yet capable of autonomously producing a consistent schematic representation of the resulting system. The final architecture diagram was therefore reconstructed from the tabular data to ensure technical accuracy, traceability, and visual consistency across all system stages.

Table 6.7.: TMS generation report - final validated system design specification

Stg.	Component Name	Function	Mass Flow (kg/s)	P (bar)	T (°C)	Valve Type & Functional.	Design Notes
1	Main Pump	Pressurize system	∼22.0	1.0 → 3.5	50 → 52	N/A	Prim.flow source, ∼75% eff.
2	Filter/Deion. Module	Coolant cleaning	22.0	3.5 → 3.3	52	N/A	Deion. & filtrat., ∼0.2 bar drop
3	Main Isolation Valve	System shutdown	22.0	3.3 → 3.2	52	Isolation-Fail-closed	Emergency shut-off capability
4	Flow Splitter	Split to branches	22.0 → (A:12.6 + B:5.5)	3.2	52	N/A	Equal pressure distribution
5a	FC Branch Flow Valve	Regulate Branch A flow	12.6	3.2 → 3.1	52	Fail-open	Precise flow for FuC stack
5b	FC Stack HX	Heat pickup (440 kW)	12.6	3.1 → 2.8	52 → 62	N/A	Primary load removal
5c	FC Branch Isolation Valve	Branch A exit	12.6	2.8 → 2.7	62	Fail-closed	Isolate for maintenance if needed
5d	DC/DC HX	Heat pickup (25 kW)	12.6	2.7 → 2.6	62 → 65	N/A	Cooling for power electronics
5e	Branch A Merge Valve	Prevent backflow	12.6	2.6 → 2.55	65	Fail-closed	Check valve to unify flow
6a	Air Cond. Branch Flow Valve	Regulate Branch B flow	5.5	3.2 → 3.1	52	Fail-open	Cabin AC flow control
6b	Air Cond. HX	Heat pickup (36.3 kW)	5.5	3.1 → 2.9	52 → 60	N/A	Cabin climate control
6c	H_2 Cond. HX	Heating load (-68.1 kW)	5.5	2.9 → 2.75	60 → 55	N/A	Warms cryogenic hydrogen
6d	Branch B Merge Valve	Prevent backflow	5.5	2.75 → 2.70	55	Fail-closed	Check valve to unify flow
7	Ram Air HX	Final heat rejection	∼18.1	2.70 → 2.40	∼ 60 → 50	N/A	Rejects up to ∼ 433 kW to amb.
8	Expansion Tank	Volume comp., conduct. control	18.1	2.40 → 2.35	50	N/A	Adds ∼0.05 bar drop, ensures fluid quality

The configuration reflects the distinct thermal requirements of the subsystems: Branch A supplies the fuel cell components with the lowest available coolant temperature (<70°C), while Branch B combines air conditioning with hydrogen conditioning, which requires heating above 60°C. A notable outcome is the coupling of H_2 conditioning with the air conditioning loop, which addresses the simultaneous demands for cooling and heating. This arrangement was derived by applying thermodynamic balance principles rather than directly following a literature precedent.

The thermodynamic evaluation of the developed TMS shows technical coherence:

- Pressure balance: Nearly all components operate within realistic pressure drop limits (0.1-0.4 bar for heat exchangers).
- Temperature constraints: All thermodynamic boundary conditions are maintained.
- Pump design: 7.22 kW electrical power for a 2.5 bar pressure increase at 22 kg/s.

Despite the high overall quality, the analysis also revealed systematic weaknesses in pressure balancing. A problematic case is the unrealistic pressure drop calculation in the return line (Stage 8: "*Expansion Tank*" at 2.35 bar to Stage 1: "*Main Pump*" at 1 bar), which was not flagged during subsequent validation. This inconsistency indicates that the LLM did not consistently reconcile interdependent parameters. For example, it failed to recognize that an unrealistically high return pressure drop would imply a correspondingly lower pump head requirement. Such cases highlight the need for human engineering supervision in verifying automated system designs.

6.5.4. Qualitative Evaluation of the Generated Architecture

The generated TMS architecture exhibits a high degree of technical completeness and internal coherence. All major heat sources and sinks were captured, the system was dimensioned within physically realistic operating ranges, and a consistent overall heat balance was achieved. A major strength of this phase lies in the explicit integration of thermodynamic constraints and embedded validation routines, which enabled the model to perform numerical reasoning based on executable code rather than heuristic approximations. This hybridization of symbolic and numerical reasoning illustrates that the methodology can generate physically coherent system architectures even in domains without established aircraft references.

Nevertheless, several limitations became apparent. The most prominent is the unrealistic pressure drop identified in the return line, which remained unreconciled with the pump sizing logic. This inconsistency indicates that the LLM did not consistently propagate dependent variables through the interconnected system equations. Moreover, redundancy strategies and the specification of auxiliary components, such as filtration and deionization modules, were addressed only in a generic manner. These shortcomings confirm that, while the methodology can ensure thermodynamic plausibility, it still lacks the iterative control loops necessary to detect and correct inconsistencies autonomously.

Addressing this limitation points toward a natural methodological extension: the introduction of *agentic refinement mechanisms* that enable iterative feedback between synthesis levels and automated self-consistency checks. In such a configuration, specialized LLM agents, for example, for requirement validation, flow analysis, and component integration, could exchange intermediate results, reconcile mismatched constraints, and verify that parameter dependencies remain consistent across system levels. This agentic layer would transform the current sequential generation pipeline into a self-evaluating, closed-loop reasoning system, substantially improving robustness and reducing the need for manual post-validation in future iterations of the SAGE workflow.

Taken together, these findings highlight that the current implementation of the SAGE methodology already produces technically coherent and physically feasible initial architectures suitable for early design exploration. While systematic engineering validation remains indispensable to verify boundary conditions, refine assumptions, and ensure robustness in safety-critical contexts, the approach demonstrates a reliable capability to generate baseline-consistent system configurations that align with engineering principles and real-world feasibility.

6.6. Summary

This second case study extended the evaluation of the SAGE methodology from an established reference domain to a technically less mature system class. While the ECS

case showed that the methodology can derive architectures converging toward a known aircraft reference system, the TMS application tested its capability to operate under near-greenfield conditions without an established reference system. Although literature on fuel cell thermal management is abundant, none of the reviewed sources addressed the required power levels and integration depth for large transport aircraft. The TMS case, therefore, served as a methodological stress test for SAGE's ability to generalize across domains and to synthesize physically coherent architectures under heterogeneous evidence.

The results confirm that the methodology remains robust and transferable even in such conditions. Recursive knowledge extraction and structured synthesis produced a comprehensive and logically consistent knowledge base, while architecture generation translated this knowledge into a thermodynamically valid and traceable system configuration. The generated TMS architecture met all major thermodynamic constraints, including heat balance, flow rates, and temperature limits, and established full traceability across the RFL chain. However, certain inconsistencies, notably the unbalanced return-line pressure drop, demonstrate that human verification remains essential for validation. The generated architecture should be interpreted as a technically plausible but preliminary configuration that forms a valid baseline for subsequent design iterations rather than a finalized system proposal.

From a methodological standpoint, three contributions distinguish the TMS case from the ECS study: (1) demonstration of robust knowledge synthesis in a fragmented domain, (2) realization of full RFL traceability from requirements to components, and (3) embedding of executable thermodynamic-based validation directly into the generative process. Together, these results show that SAGE can scale from the reproduction of known systems to the creation of novel architectures in unexplored design spaces.

Looking forward, the observed limitations point toward an evolution of the methodology toward *agentic architectures*. Future iterations of SAGE could employ multiple cooperating LLM agents that autonomously exchange intermediate results, validate parameter consistency, and dynamically reconcile discrepancies between abstraction levels. Such systems, which are already emerging in the research fields of multi-agent reasoning and tool-using LLMs, would extend the current sequential workflow into a closed, self-correcting design loop. By combining synthesis, validation, and reasoning agents in an orchestrated framework, SAGE could evolve from a structured generative methodology into a distributed co-engineering environment that continuously optimizes its own outputs.

Taken together, the ECS and TMS case studies confirm that SAGE scales from reproducing established reference systems to generating technically plausible architectures in emerging domains. They jointly demonstrate that the methodology can support engineers across different stages of conceptual design: from systematically structuring known system knowledge to developing initial concepts in areas where only small-scale precedents exist. Collectively, the results provide a proof-of-concept for the SAGE methodology and lay the groundwork for its further refinement and integration into model-based and GenAI-augmented systems engineering workflows.

7. Discussion

The preceding chapters introduced, implemented, and empirically evaluated the SAGE methodology. While these results demonstrate its feasibility and practical usefulness, the broader methodological significance emerges only from a higher-level analysis. This chapter, therefore, shifts the perspective from step-by-step development and case-study implementation to an overarching reflection on SAGE's performance and implications.

Its objectives are to synthesize the central methodological insights, assess methodological efficiency, and situate SAGE within the landscape of established development approaches. In doing so, the discussion synthesizes the empirical results with the overarching questions regarding the benefits, boundaries, and implications of GenAI in Systems Engineering.

7.1. Iteration Time and Workload

The two case studies demonstrate that the SAGE methodology achieves substantial time and resource savings compared to conventional development approaches. The decisive difference lies in the automation of knowledge-intensive tasks: SAGE performs the majority of information retrieval, structuring, and synthesis autonomously. As a result, the role of engineers would shift from manual document analysis toward configuration, monitoring, and interpretation of the generated results.

The most time-consuming activity in both studies was the acquisition of literature, particularly the manual downloading of individual papers. In the ECS case, this step (covering 11 documents) took roughly 30 minutes for the download process alone, while the TMS case with 115 documents required a proportionally higher effort. Subsequent data preparation, including cleaning, segmentation, and formatting, was mainly handled automatically by the preprocessing pipeline. Overall, the total effort for document acquisition scales almost linearly with the number of sources, although it also depends on their availability and database accessibility.

Once the data are collected, the automated processing pipeline performs the majority of the work. Across both case studies, the total processing time from raw documents to a completed architecture report was around 10-12 hours. Most of this time was spent in Phase 1 (document preprocessing, approx. 3 hours) and Phase 2 (recursive knowledge extraction, approx. 6 hours). The duration of Phase 2 mainly depends on the number of questions to be answered: around six hours were required to answer 930 questions. Phase 3 (knowledge synthesis) and Phase 4 (architecture generation) roughly needed 0.5-1 hour each, including brief manual interventions for prompt adjustments and validation. Consequently, the effective manual workload for producing complete conceptual architectures amounts to only 1-2 hours, while the remaining steps run fully autonomously.

The cost structure differs fundamentally from that of traditional systems engineering. Whereas conventional processes are dominated by personnel costs accumulated over extended time periods, SAGE primarily incurs predictable API expenses for LLM usage. In the present case studies, these costs remained negligible. Even for large-scale industrial applications involving thousands of documents, the resulting computational costs would likely remain marginal compared to the corresponding reduction in engineering labor.

Another influential factor is the document type. Scientific publications and patents, used in both case studies, require extensive preprocessing, as relevant information is often implicit and fragmented over different documents. In industrial environments, by contrast, requirements, interface definitions, and functional specifications are typically stored in a structured digital form. Under such conditions, the pipeline can be expected to operate more efficiently, with higher semantic precision and reduced preprocessing effort.

In summary, SAGE reduces the time required to generate initial architectures from weeks or months to only a few hours of combined manual and computational effort. This shift redefines the focus of engineering work, moving from laborious document processing toward the critical evaluation, validation, and refinement of automatically generated results. Rather than replacing engineering activities, SAGE transforms them into a process in which information handling is automated, allowing engineers' effort to be devoted to reasoning, evaluation, and design decisions. In practice, this transformation also changes the cognitive load of engineering work. Reviewing and refining a GenAI-generated report often proves easier and more effective than developing one entirely from scratch, as engineers can focus on identifying weaknesses, inconsistencies, or design risks rather than constructing every element themselves. Similar to reviewing a scientific paper, it is often easier to identify gaps and potential pitfalls in an existing draft than to create a complete and error-free version from scratch. By enabling this shift, SAGE enhances both efficiency and design quality while preserving the essential role of human expertise.

7.2. Classification of SAGE within Existing Development Methodologies

The development of technical systems can follow various methodologies, each emphasizing distinct approaches to representing and managing system knowledge. This section situates SAGE within this landscape by comparing the traditional document-based approach, Model-Based Systems Engineering (MBSE), and the emerging GenAI-supported methodology. The comparison focuses on their respective treatment of information, effort, and scalability, highlighting how SAGE complements existing frameworks rather than replacing them.

Document-Based Systems Engineering (DBSE):
The traditional document-based approach follows the systematic RFLP process through iterative document creation. Independent documents are created for each development step, which must be manually created, reviewed, and coordinated between individual phases. This methodology is characterized by iterative workflows where changes in early phases require costly adjustments in downstream documents. Consistency between different documents must be continuously manually checked and ensured, which brings considerable coordination and time effort. Time periods of several months are realistic for complete concept development, with actual duration heavily dependent on the system under consideration, team size, organizational structure, and available expertise.

Model-Based Systems Engineering (MBSE):
MBSE replaces document-centric approaches with structured domain models that act as the central medium for system representation and information exchange. The methodology uses formal modeling languages like SysML for systematic capture of requirements, functions, and architectures in a coherent digital model. A qualitative consideration of MBSE implementation shows a characteristic pattern of high initial investments with long-term efficiency gains. The integration of MBSE into established Systems Engineering organizations continues to present considerable challenges, as it necessitates substantial modifications to traditional workflows and organizational practices.

SAGE as Complement:
The SAGE methodology is neither a replacement for DBSE nor for MBSE, but a complementary extension. It particularly addresses those challenges that both approaches share: the time-intensive processing of large knowledge bases and the generation of consistent foundations for architecture development. SAGE automates the extraction and structuring of relevant content from extensive document corpora and organizes it according to RFLP logic. The methodology thus enables rapid, objective, and source-traceable consolidation of existing knowledge.

Its strengths lie in systematic domain exploration and transparent documentation of the knowledge base. SAGE thus creates a bridge between document-based and model-based system development: it can accelerate DBSE processes by reducing manual document review and preparation, and serve as a precursor to MBSE by providing consolidated knowledge structures that can be transferred into formal models.

7.3. Advantages and Limitations of the SAGE Methodology

The two case studies confirm that SAGE particularly excels in domains that involve large volumes of textual information. At the same time, its impact naturally varies with the complexity and structure of the underlying data.

A key advantage lies in automating knowledge-intensive work steps. In particular, the phases of recursive knowledge extraction (Phase 2) and structured synthesis (Phase 3) proved to be reliable and reproducible. In both the ECS and TMS case studies, the essential requirements, functions, and components were successfully identified and classified according to the RFLP* logic. It is noteworthy that these results were derived exclusively from scientific publications and patents, where relevant knowledge was often only implicitly contained. When applied to structured industrial data such as requirement lists, function catalogs, or system models, the methodology can therefore be expected to operate with even higher efficiency and accuracy.

SAGE demonstrates particular strength in text-based tasks. The methodology can systematically structure large volumes of documentation, ensure consistent formatting, and maintain traceability across all extracted and synthesized information. This applies both to the identification of individual knowledge elements and to their integration into a consolidated RFLP*-based document. Such results provide immediate practical value for reviews, compliance demonstrations, and the preparation of certification documents, as they enhance transparency and significantly reduce the effort required for manual consistency checks. Beyond these immediate benefits, the methodology offers long-term advantages through its modular and adaptive structure. By expanding the document corpus, refining prompt strategies, or integrating improved models, the process can be continuously enhanced without restarting from scratch. At the same time, SAGE contributes to knowledge preservation and transfer by converting unstructured textual information into structured, reusable knowledge bases that facilitate organizational learning and cross-project reuse. Economically, the compression of early development phases translates directly into cost efficiency: shorter iteration cycles, reduced manual effort, and earlier identification of viable design directions minimize downstream changes and enable engineering resources to be redirected toward innovation.

By contrast, the architecture generation results must be interpreted more cautiously. In both case studies, SAGE produced technically coherent architectures that serve as plausible conceptual starting points. However, these remained at a high level of abstraction and

were not yet optimized, validated, or ready for implementation. The main value lies in the preparatory groundwork: SAGE consolidates the knowledge base on which engineers can subsequently perform detailed and optimized design work. The limitations of the methodology thus arise less from technical shortcomings than from intrinsic task characteristics and the current state of LLM technology. Output quality depends directly on the completeness and accuracy of the document base, as incomplete or outdated information can propagate through the process. Optimal input data is therefore characterized by explicit, domain-specific content in a structured form, such as requirement lists or function catalogs, rather than information that is only implicitly contained within narrative text. Moreover, while LLMs are effective at identifying and recombining knowledge patterns, they still fall short of the mathematical precision and optimization depth required for industrial design. Finally, the effective use of SAGE requires close engineer-GenAI collaboration. The methodology reaches its full potential only when engineers critically review, refine, and validate the generated results, demanding new skills in prompt formulation, model interpretation, and AI-assisted design reasoning.

In essence, SAGE provides substantial advantages in the systematic preparation, structuring, and consolidation of engineering knowledge. However, it does not replace the creative, optimizing, or validating tasks performed by engineers. Instead, it establishes a new division of labor: the system automates information handling, allowing human expertise to focus on reasoning, evaluation, and innovation. In this way, SAGE transforms the engineering process from one of manual knowledge acquisition to one of guided, critical, and collaborative synthesis.

7.4. Industry Impact

The two case studies indicate that the SAGE methodology holds significant potential not only for research but also for industrial applications. Even when based solely on publicly available sources with largely implicit information, SAGE was able to identify consistent requirements, functions, and architectural elements. When applied to real industrial environments, where documentation is typically explicit and structured, its benefits are expected to increase substantially.

In practical terms, the most significant advantages emerge in three areas:

- Document processing and consolidation: SAGE reduces the effort for reviewing, structuring, and consistency checking of extensive document collections. This applies not only to external literature, but especially to internal company documents, whose manual evaluation currently binds considerable resources.
- Transparency and traceability: Through systematic source documentation, SAGE supports internal reviews and regulatory evidence. Especially in the aircraft industry with its strict certification requirements, this is a decisive added value.
- Support in early development phases: SAGE can deliver consistent and traceable pre-architectures that serve as structured starting points for further development. While these concepts are not finalized solutions, they provide a coherent framework from which engineering teams can efficiently continue their design work.

Overall, SAGE has the potential to accelerate the knowledge-intensive stages of system development substantially. Its primary value lies in reducing repetitive documentation tasks and in generating transparent, consistent knowledge foundations. The methodology does not diminish the responsibility of engineers but rather augments their capabilities, enabling a shift of focus from information retrieval toward creative, optimizing, and validating engineering activities.

8. Conclusion and Outlook

The development of aircraft models and their systems follows the structured RFLP logic of systems engineering: requirements are systematically mapped to functions, which are then translated into logical structures and ultimately into physical components. This framework forms the backbone of industrial practice, ensuring that technical systems can be designed in a consistent and traceable manner. At the same time, aircraft development is characterized by a high level of effort in knowledge processing. New aircraft models or their variants are rarely created from scratch. Instead, they begin by analyzing which systems, subsystems, components, or requirements can be reused, which must be adapted, and which require new development. This so-called reuse/adapt/new analysis is indispensable, highly iterative, and, due to the need to search through extensive, often decentralized data stores, extremely resource-intensive.

The central question of this dissertation is whether GenAI can help systematize and accelerate the key steps in aircraft system development, namely knowledge extraction, reuse/adapt/new analysis, as well as structuring the document along RFLP logic. This motivation arises from two converging trends: on the one hand, aircraft programs face ever-increasing pressure from extensive documentation, long development cycles, and shortages of qualified engineers. On the other hand, GenAI technologies have rapidly matured from research prototypes into widely adopted industrial tools, offering unprecedented capabilities in large-scale text analysis and knowledge processing.

Large language models (LLMs) can process extensive technical texts and provide contextual answers, but they are constrained by several factors: the limited context window, the tendency to hallucinate without explicit source grounding, and the lack of direct access to external or internal knowledge bases[1]. Retrieval-Augmented Generation (RAG) addresses some of these shortcomings by enriching model prompts with retrieved, source-linked evidence. Through this mechanism, language models can be connected to curated document repositories, ensuring that generated outputs are anchored in verifiable knowledge. In practical applications, such repositories may comprise both external sources, such as scientific publications or regulatory standards, and internal corporate knowledge bases, such as requirements databases or system models. However, conventional RAG systems are primarily designed for isolated query-answer interactions. Each question is answered in isolation, without building a deeper and more systematic understanding of the interconnected technical relationships.

Building on this, the SAGE (*System Architecture through Generative Engineering*) methodology was developed to combine the reasoning and text analysis capabilities of LLMs with the reliability and traceability of RAG, while overcoming the single-query limitation. SAGE creates a structured question-answer database that is compact enough to fit into the context window of some LLMs, while maintaining a secure link to its sources. This recursive and systematic integration makes the use of GenAI in systems engineering both practical and traceable.

[1]The context window refers to the maximum text length that a language model can process at one time. If this is exceeded, information is lost, or prompt execution does not take place at all. Hallucination refers to the property of generative models to produce seemingly plausible but factually incorrect content, especially when no reliable sources are available.

The methodological development of SAGE occurred in four clearly delineated phases, each addressing different questions and building upon the others. In Phase 1, document preparation was the focus. Sources such as scientific publications and patents were transformed into a uniform, machine-readable format. This step was necessary to create a consistent foundation for subsequent analyses.

In Phase 2, recursive knowledge extraction was developed, which forms the core of the methodology. While conventional RAG operates on the principle of "one question - one answer," SAGE methodically takes a different approach. As demonstrated in Chapter 2.4.3, previous research has already recognized that some questions require sub-questions before they can be answered correctly. A typical example is the question "*Where did the physicist who developed the theory of relativity study?*" First, the physicist (Albert Einstein) has to be identified before the university (ETH Zurich) can be determined. This idea of decomposing questions into sub-questions was adopted in SAGE. Not to refine a single answer, but rather to build up a systematically growing knowledge base. Each question-answer pairing is stored, supplemented with new sub-questions, and thus expanded into a systematically growing knowledge base. Instead of providing a specific answer, a knowledge base is created that can subsequently be used for architectural development.

In Phase 3, this knowledge base was consolidated and organized along the RFLP structure into system context, requirements, functions, logical elements, physical components, and architectural concepts. The central question was whether it was possible to extract complex system descriptions from the literature and structure them systematically. This experiment proved to be particularly insightful. Even when relevant content was only implicitly contained in the sources, for instance, in narrative texts without explicit requirement lists, it was still possible to build consistent structures. Phase 3 was therefore not an optional intermediate step, but rather a methodological safeguard to ensure the reliable synthesis of knowledge from publicly available texts. Only on this foundation could the decisive question in Phase 4 be investigated.

Phase 4 initially focused on the reuse/adapt/new analysis. The aim was to check whether the consolidated knowledge bases could reliably distinguish between existing elements, elements to be adapted, and elements to be newly developed. The results confirmed that SAGE can reliably support this critical task. Following this, the methodology was deliberately further challenged. It was tested whether complete system architectures could also be generated based on the extracted information. The results showed that consistent architectures were generated, but that they remained at a conceptual level of abstraction. While they are suitable as plausible starting points for engineering teams, they do not represent industry-ready end solutions. However, it is noteworthy that the documentation of requirements, functions, logical elements, and components worked very reliably. This part of the analysis proved particularly robust and consistent. Limitations only became apparent in the final step of the design process, which, in the context of an iterative approach, is to be expected as a preliminary draft.

These four phases clarify why the methodology was developed in this form. Each phase builds on the previous one. For example, without ensuring in Phase 3 that complex information can be structurally consolidated, the Reuse/Adapt/New analysis would not have been reliable. Likewise, Phase 4 could only be meaningfully conducted because the reliability of knowledge synthesis had been previously demonstrated. This demonstrates that SAGE's strength lies not in the direct creation of final architectures, but in methodically validating the steps that lead to them.

The two case studies have demonstrated that SAGE can accelerate the development of aircraft systems in those phases that rely heavily on text information. The methodology proved particularly effective in systematic knowledge extraction and classification along RFLP logic, as well as in supporting reuse, adaptation, and new analyses. However, it also

became clear that engineers continue to play a central role in creative optimization, precise dimensioning, and the validation of safety-critical systems. The result is a process in which automated knowledge processing and human evaluation complement each other.

Methodologically, this dissertation offers numerous starting points for further developments. The recursive generation of sub-questions could be combined in the future with multi-hop or graph-based methods to increase coverage further and improve the quality of answers. Similarly, the integration of automated validation mechanisms offers particularly high potential. Automated statement validation and confidence classification refinement represent a key development step in this regard. Generated statements could be systematically checked against the original RAG-based question-answer pairs, with the classification system being dynamically adapted to the results. This would substantially enhance the methodological reliability. This approach addresses a key limitation of the present work, namely the manual evaluation of individual statements, and would make the methodology considerably more robust for industrial applications with larger data inventories.

The possibilities of closer connection to MBSE tools appear particularly promising. Initial investigations into coupling with SysML v2 already indicate that the knowledge bases generated in SAGE can be directly converted into formal models. This would significantly enhance continuity in the development process by automating the transition from text-based knowledge synthesis to model-based system representations. Another direction for future research concerns the optimization of configuration parameters. In this dissertation, local optima could be identified (e.g., for the retrieval value k, the choice of LLM, and modular prompt structures), but the complex interdependencies between RAG extraction and LLM synthesis were not systematically investigated. A multivariate parameter optimization that takes into account the interactions of all control variables could reveal global strategies for improving performance across all phases.

Finally, the integration of proprietary knowledge bases presents considerable potential, but also poses challenges that should not be underestimated. While this work was based on publicly accessible sources that frequently contained only implicit information, more explicit and more strongly structured documents, such as complete requirement catalogs, function descriptions, and validated architectures, are typically available in industrial contexts. It is plausible to assume that the methodology could work even more efficiently in such an environment, since less reconstruction of implicit content would be required. Future research should therefore focus on secure implementations that enable the processing of confidential documents without compromising the protection of intellectual property. Such integration would not only increase the efficiency of the methodology but simultaneously significantly increase its practical relevance for industrial applications.

Research is currently focusing more and more on agentic architectures. Several different approaches are emerging in this area. One line of development deals with agents that integrate external tools or specialized modules, for example, for calculations, databases, or simulations, and dynamically coordinate their results. Concepts corresponding to this approach are already in use in multi-agent systems, tool-using LLMs, and planning agents. These systems demonstrate how LLMs can integrate external programs and data sources into complex workflows. In parallel, another trend has emerged in the scientific literature known as “Agentic RAG”. For instance, Lelong et al. [207] present INRAExplorer, a system in which an LLM acts as an orchestrator between knowledge graph queries, publication retrieval, and expert identification to answer complex multi-hop questions. Liang et al. [208] systematize reasoning approaches in a survey based on dual-process theory, distinguishing between predefined pipelines and agentic methods, in which the LLM independently decides on retrieval. In addition, Li et al. [209] show how reasoning and retrieval can be mutually reinforced in synergized frameworks. While these works differ from the SAGE methodology developed here, they clarify the field’s direction away from isolated answers and toward structured, iterative, and domain-specific knowledge

processes. The fact that these publications appeared after April 2025 underscores the timeliness and relevance of the approach presented here.

In summary, this dissertation presents a proof-of-concept that demonstrates how GenAI can be systematically integrated into aircraft systems engineering. The developed SAGE methodology establishes a structured, traceable, and scalable framework for generative systems engineering, combining extended RAG and LLM reasoning. While RAG enables structured and source-linked access to verified domain knowledge, LLM reasoning provides the semantic interpretation, synthesis, and transformation of this information into consistent engineering representations. In doing so, SAGE bridges the gap between automated knowledge retrieval and structured system design. The results confirm that SAGE performs exceptionally well in structuring and synthesizing complex technical knowledge along RFLP logic, enabling the systematic reconstruction of system relationships that would otherwise require substantial manual effort. At the same time, limitations become evident in mathematical modeling and detailed architectural optimization, where engineering expertise remains indispensable. The methodology thus achieves a new division of labor: automation, where knowledge can be formalized, and engineering judgment, where creativity and precision are required.

Overall, SAGE demonstrates the transformative potential of combining GenAI with established systems engineering principles. It accelerates the early development phases by creating a structured, traceable, and scalable foundation for informed design decisions. While its application in this dissertation was limited to publicly available data and conceptual architectures, the approach provides a solid basis for further development and industrial adaptation. Beyond the aircraft domain, the methodology opens perspectives for transfer to other complex engineering fields, where systematic knowledge synthesis and reuse are equally critical. As GenAI technologies continue to mature, approaches such as SAGE are likely to become essential accelerators in the transition toward faster, more consistent, and knowledge-driven engineering.

Bibliography

[1] Voth, V., Lübbe, S., and Bertram, O.: "Estimating Aircraft Power Requirements: A Study of Electrical Power Demand Across Various Aircraft Models and Flight Phases". In: *Aerospace* 11(12) (2024), p. 958. DOI: 10.3390/aerospace11120958.

[2] Voth, V. and Bertram, O.: "Aircraft System Design: A Model-Based and Collaborative Approach". In: *ICAS Proceedings 34th International Council of the Aeronautical Spaces, September 2024, Florence (Italy)*. 2024. URL: https://www.icas.org/icas_archive/icas2024/data/papers/icas2024_0160_paper.pdf (visited on 07/09/2025).

[3] Voth, V., Lübbe, S., Schäfer, M., et al.: "Functional Approach to a Fuel Cell Thermal Management System in Safety-Critical Applications". In: *AIAA Aviation 2023 Forum, June 2023, San Diego (USA)*. Reston, Virginia: American Institute of Aeronautics and Astronautics, 2023. DOI: 10.2514/6.2023-3879.

[4] Lübbe, S., Voth, V., Schäfer, M., et al.: "Interconnections in Model-Based Safety Analysis and Systems Design on the Example of a Fuel Cell Thermal Management System for Commercial Aircraft". In: *AIAA Aviation 2023 Forum, June 2023, San Diego (USA)*. Reston, Virginia: American Institute of Aeronautics and Astronautics, 2023. DOI: 10.2514/6.2023-4196.

[5] Schorr, M., Voth, V., and Gentner, C.: "Effects on the design of aeronautical fuel cell systems by inclusion of reliability requirements". In: *CEAS Aeronautical Journal* 15(4) (2024), pp. 807–820. DOI: 10.1007/s13272-024-00743-9.

[6] Schorr, M., Voth, V., and Gentner, C.: "Auswirkungen von Redundanzanforderungen auf den Entwurf von Brennstoffzellen-Systemen in Verkehrsflugzeugen". In: *Deutscher Luft- und Raumfahrtkongress, September 2022, Dresden (Germany)*. 2022.

[7] Voth, V., Wade, J., Bertram, O., et al.: "Evaluation of a Generative AI Methodology for Technical System Development: A Case Study of an Aircraft Environmental Control System". In: *AIAA SciTech 2026 Forum, January 2026, Orlando (USA)*. Reston, Virginia: American Institute of Aeronautics and Astronautics, 2026.

[8] Winter, C.: "A Way To Improve Profitability Of Revolutionary Civil Aircraft Projects". In: *Kwartalnik Naukowy Uczelni Vistula* 3(49) (2016), pp. 152–172.

[9] Airbus: *Airbus A380*. URL: https://www.airbus.com/en/products-services/commercial-aircraft/passenger-aircraft/a380 (visited on 09/21/2025).

[10] Zhao, Y.: "Why 787 delays were inevitable?" In: *SSRN* (2015). DOI: 10.2139/ssrn.2647588.

[11] International Organization for Standardization (ISO), International Electrotechnical Commission (IEC), and Institute of Electrical and Electronics Engineers (IEEE): *Systems and software engineering - System life cycle processes*. ISO/IEC/IEEE 15288:2023. 2023. URL: https://ieeexplore.ieee.org/servlet/opac?punumber=10123345 (visited on 06/01/2025).

[12] INCOSE: *Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities*. 4th ed. New York: John Wiley & Sons, 2015. ISBN: 978-1118999400.

[13] Pardessus, T.: "Concurrent Engineering Development and Practices for Aircraft Design at Airbus". In: *ICAS Proceedings 24th International Congress of the Aeronautical Sciences, September 2004, Yokohama (Japan)*. 2004. URL: https://www.icas.org/icas_archive/ICAS2004/PAPERS/413.PDF (visited on 07/09/2025).

[14] Snowden, D. J. and Boone, M. E.: "A Leader's Framework for Decision Making". In: *Harvard Business Review* 85(11) (2007), pp. 68–76. URL: https://hbr.org/2007/11/a-leaders-framework-for-decision-making (visited on 07/09/2025).

[15] Ingenium - Canada Aviation And Space Museum: *Douglas DC-3*. URL: https://ingenium.ca/aviation/en/collection-highlight/douglas-dc-3/ (visited on 07/24/2023).

[16] Robinson, D.: *Lockheed Constellation*. South Australian Aviation Museum. Oct. 2015. URL: https://www.saam.org.au/history_group_docs/SAAM%20Aircraft%20-%20LOCKHEED%20Constellation.pdf (visited on 07/24/2023).

[17] Shah, D.: *Millions of data points flying in tight formation*. Aerospace Manufacturing and Design. 2014. URL: https://www.aerospacemanufacturinganddesign.com/news/millions-of-data-points-flying-part2-121914/#:~:text=For%20the%20past%20several%20years%2C,These%20systems%20have%20provided (visited on 05/31/2025).

[18] Sinnett, M.: "787 No-Bleed Systems: Saving Fuel and Enhancing Operational Efficiencies". In: *Aero Quaterly*. Vol. 18. 2007, pp. 6–11.

[19] Kowalski, R.: "Drehgeberlose Regelung elektromechanischer Flugsteuerungsaktuatoren". PhD thesis. Braunschweig: Technische Universität Braunschweig, 2024. DOI: 10.57676/hcb3-9c65.

[20] Itier, J.-B.: "A380 integrated modular avionics". In: *ARTIST2 project meeting on integrated modular avionics*. Vol. 1. 2. 2007, pp. 72–75. URL: http://www.artist-embedded.org/docs/Events/2007/IMA/Slides/ARTIST2_IMA_Itier.pdf (visited on 09/15/2025).

[21] Merino-Martinez, R., Besnea, I., Hoff, B. von den, et al.: "Psychoacoustic analysis of the noise emissions from the Airbus A320 aircraft family and its nose landing gear system". In: *30th AIAA/CEAS Aeroacoustics Conference, June 2024, Rome (Italy)*. 2024, p. 3398. DOI: 10.2514/6.2024-3398.

[22] Zhongda, Y. and Mingguang, L.: "Maintenance differences between 737MAX and 320neo". In: *4th International Conference on Applied Materials and Manufacturing Technology, May 2018, Nanchang (China)*. Vol. 423. 1. IOP Publishing, 2018. DOI: 10.1088/1757-899X/423/1/012059.

[23] Steinke, S.: *Erster Airbus A320neo im Liniendienst*. FlugRevue. 2024. URL: https://www.flugrevue.de/zivil/neo-premiere-erster-airbus-a320neo-im-liniendienst/ (visited on 04/16/2016).

[24] Spitz, W., Berardino, F., Golaszewski, R., et al.: *Development cycle time simulation for civil aircraft*. Tech. rep. National Aeronautics and Space Administration, 2001.

[25] Costa Campos, L. M. B. da: *Accelerating Sustainable Aviation Initiatives: Technology, Markets and Social Issues*. Springer Cham, 2025. DOI: 10.1007/978-3-031-73993-4.

[26] Hunt, A.: *Airbus A320 single-aisle Aircraft*. AviationHunt. 2024. URL: https://www.aviationhunt.com/airbus-a320/ (visited on 02/11/2024).

[27] Torenbeek, E.: *Advanced aircraft design: Conceptual design, analysis, and optimization of subsonic civil airplanes*. Aerospace series. West Sussex United Kingdom: John Wiley & Sons Ltd., 2013. DOI: 10.1002/9781118568101.

[28] SAE International Recommended Practice: *Guidelines for Development of Civil Aircraft and Systems*. SAE Standard ARP4754B. Revised December 2023, Issued November 1996. DOI: https://doi.org/10.4271/ARP4754B.

[29] Tschirner, U.: *Zwei Drittel der Maschinenbauer klagen über Mangel an Ingenieuren*. Die Zeit. 2022. URL: https://www.zeit.de/arbeit/2022-11/ingenieure-offene-stellen-vdma-maschinenbau (visited on 11/03/2023).

[30] Klinkusch, J.: "Demografischer Wandel: Nachwuchsmangel bei Ingenieuren - was zu tun ist". In: *Verein Deutscher Ingenieure* (2019). URL: https://www.vdi.de/news/detail/demografischer-wandel-nachwuchsmangel-bei-ingenieuren-was-zu-tun-ist (visited on 05/29/2025).

[31] Statistisches Bundesamt: *Wintersemester 2023/2024: Studierendenzahl weiter rückläufig: Pressemitteilung Nr. 455 vom 28. November 2023*. 2023. URL: https://www.destatis.de/DE/Presse/Pressemitteilungen/2023/11/PD23_455_21.html (visited on 05/05/2024).

[32] Heublein, U., Hutzsch, C., and Schmelzer, R.: "Die Entwicklung der Studienabbruchquoten in Deutschland". In: Deutsches Zentrum für Hochschul- und Wissenschaftsforschung (DZHW)(5) (2022). DOI: 10.34878/2022.05.dzhw_brief.

[33] Brown, T., Mann, B., Ryder, N., et al.: "Language models are few-shot learners". In: *Proceedings of the 34th International Conference on Neural Information Processing Systems(NeurIPS), December 2020, Canada*. Ed. by Larochelle, H., Ranzato, M., Hadsell, R., et al. 2020, pp. 1877–1901. DOI: 10.48550/arXiv.2005.14165.

[34] OpenAI: *Introducing ChatGPT*. 2022. URL: https://openai.com/index/chatgpt/ (visited on 05/24/2025).

[35] Krus, P.: "Large Language Model in Aircraft System Design". In: *ICAS Proceedings 34th Congress of the International Council of the Aeronautical Sciences, September 2024, Florence (Italy)*. 2024. URL: https://www.icas.org/icas_archive/icas2024/data/papers/icas2024_0514_paper.pdf (visited on 06/14/2025).

[36] Buchholz, K.: *Threads Shoots Past One Million User Mark at Lightning Speed*. Statista. 2023. URL: https://www.statista.com/chart/29174/time-to-one-million-users/ (visited on 07/14/2025).

[37] Singla, A., Sukharevsky, A., Yee, L., et al.: *The state of AI: How organizations are rewiring to capture value*. McKinsey & Company. 2025. URL: https://www.mckinsey.com/capabilities/quantumblack/our-insights/the-state-of-ai#/ (visited on 07/16/2025).

[38] MIT Technology Review: *10 Breakthrough Technologies*. 2025. URL: https://www.technologyreview.com/2025/01/03/1109178/10-breakthrough-technologies-2025/ (visited on 05/24/2025).

[39] Bubeck, S., Chandrasekaran, V., Eldan, R., et al.: "Sparks of artificial general intelligence: Early experiments with gpt-4". In: *arXiv preprint arXiv:2303.12712* (2023). DOI: 10.48550/arXiv.2303.12712.

[40] Kumar, P.: "Large language models (LLMs): survey, technical frameworks, and future challenges". In: *Artificial Intelligence Review* 57(10) (2024), p. 260. DOI: 10.1007/s10462-024-10888-y.

[41] Martineau, K.: *Whats an LLM context window and why is it getting larger?* IBM. 2024. URL: https://research.ibm.com/blog/larger-context-window? (visited on 09/03/2025).

[42] Solanki Shivam R.; Khublani, D. K.: *Generative Artificial Intelligence: Exploring the Power and Potential of Generative AI*. 1st edition. Apress Berkeley, CA, 2024. DOI: 10.1007/979-8-8688-0403-8.

[43] IONOS: *Retrieval-Augmented Generation (RAG) - Was ist das?* IONOS Digital Guide. 2024. URL: https://www.ionos.de/digitalguide/server/knowhow/retrieval-augmented-generation/ (visited on 07/17/2024).

[44] Gräßler, I. and Oleff, C.: *Systems Engineering*. Springer Vieweg Berlin, Heidelberg, 2022. DOI: 10.1007/978-3-662-64517-8.

[45] INCOSE: *Systems Engineering Vision 2035: Engineering Solutions For A Better World*. URL: https://www.incose.org/docs/default-source/se-vision/incose-se-vision-2035.pdf?sfvrsn=e32063c7_10 (visited on 03/20/2025).

[46] SEBoK: *Guide to the Systems Engineering Body of Knowledge (SEBoK), version 2.12*. 2025. URL: https://sebokwiki.org/w/images/e/e1/Guide_to_the_Systems_Engineering_Body_of_Knowledge_v2.12.pdf (visited on 08/12/2025).

[47] Eigner, M., Roubanov, D., and Zafirov, R.: *Modellbasierte virtuelle Produktentwicklung*. Springer Vieweg Berlin, Heidelberg, 2014. DOI: 10.1007/978-3-662-43816-9.

[48] Eigner, M., Dickopf, T., and Apostolov, H.: "The Evolution of the V-Model: From VDI 2206 to a System Engineering Based Approach for Developing Cybertronic Systems". In: *Product Lifecycle Management and the Industry of the Future*. Ed. by Ríos, J., Bernard, A., Bouras, A., et al. Vol. 517. Springer International Publishing, 2017, pp. 382–393. DOI: 10.1007/978-3-319-72905-3_34.

[49] National Aeronautics and Space Administration: *NASA Systems Enginering Handbook*. 2016. URL: https://www.nasa.gov/sites/default/files/atoms/files/nasa_systems_engineering_handbook_0.pdf (visited on 09/21/2023).

[50] Sales, D. and Becker, L. B.: "Systematic Literature Review of System Engineering Design Methods". In: *2018 VIII Brazilian Symposium on Computing Systems Engineering (SBESC), November 2018, Brazilian*. IEEE, 2018, pp. 213–218. DOI: 10.1109/SBESC.2018.00040.

[51] Weilkiens, T., Lamm, J. G., Roth, S., et al.: *Model-Based System Architecture*. 2nd ed. John Wiley & Sons, 2016. DOI: 10.1002/9781119746683.

[52] Gräßler, I., Hentze, J., and Bruckmann, T.: "V-Models for Interdisciplinary Systems Engineering". In: *Proceedings of the DESIGN 2018 15th International Design Conference, May 2018, Dubrovnik (Croatia)*. 2018, pp. 747–756. DOI: 10.21278/idc.2018.0333.

[53] Hintze, H., Wiegraefe, B., and God, R.: "A security engineering process approach for the future development of complex aircraft cabin systems". In: *IFIP International Information Security Conference, July 2013, Auckland (New Zealand)*. Springer. 2013, pp. 190–202. DOI: 10.1007/978-3-642-39218-4.

[54] Akundi, A. and Lopez, V.: "A Review on Application of Model Based Systems Engineering to Manufacturing and Production Engineering Systems". In: *Procedia Computer Science* 185 (2021), pp. 101–108. DOI: 10.1016/j.procs.2021.05.011.

[55] Ahlbrecht, A.: "Erweiterung von MBSE Prozessen bei der Entwicklung sicherheitskritischer Systemarchitekturen durch die Nutzung Formaler Methoden". Master thesis. Braunschweig: Technische Universität Braunschweig, 2021.

[56] Schummer, F. and Hyba, M.: "An approach for system analysis with model-based systems engineering and graph data engineering". In: *Data-Centric Engineering* 3 (2022). DOI: 10.1017/dce.2022.33.

[57] Arnould, V.: "Using model-driven approach for engineering the System Engineering System". In: *Proceedings of the 13th Annual Conference on System of Systems Engineering (SoSE), June 2018, Paris (France)*. IEEE, 2018, pp. 608–614. DOI: 10.1109/SYSOSE.2018.8428769.

[58] Hoffmann, M.: "Model-Based Systems Engineering for Offshore HVDC Grid Integration Systems". PhD thesis. Braunschweig: Technische Universität Braunschweig, 2021.

[59] Vosgien, T.: "Model-based system engineering enabling design-analysis data integration in digital design environments: application to collaborative aeronautics simulation-based design process and turbojet integration studies". PhD thesis. Paris: Ecole Centrale Paris, 2015.

[60] Powelske, J. A.: "MBSE-gestützte Methoden zur Strukturierung und Anwendung von Baukästen in der Frühen Phase der PGE - Produktgenerationsentwicklung mechatronischer Steuergeräte im Einsatz in Kleinantrieben im Automobilbereich". PhD thesis. Karlsruhe: Karlsruher Institut für Technologie, 2023.

[61] Cloutier R.; Bone, M.: "MBSE Survey". In: *INCOSE IW, January 2015, Los Angeles (USA)*. 2015. URL: https://www.omgwiki.org/MBSE/lib/exe/fetch.php?media=mbse:incose_mbse_survey_results_initial_report_2015_01_24.pdf (visited on 07/09/2025).

[62] Scherer, H.: "Modellbasierte Methoden zur Modellierung des Zielsystems und des Funktions-Gestalt-Zusammenhangs zur Unterstützung der Serienentwicklung von Baukästen am Beispiel von Hybrid-Triebstrangsystemen". PhD thesis. Karlsruhe: Karlsruher Institut für Technologie, 2016.

[63] Alt, O.: *Modellbasierte Systementwicklung mit SysML*. München: Carl Hanser Verlag, 2012. ISBN: 978-3-446-43066-2.

[64] Bursac, N.: "Model Based Systems Engineering zur Unterstützung der Baukastenentwicklung im Kontext der Frühen Phase der Produktgenerationsentwicklung". PhD thesis. Karlsruhe: Karlsruher Institut für Technologie, 2016.

[65] Zhang, J. A., Bagdatli, B., and Mavris, D. N.: "Leveraging SysML V2 for Integration of MBSE and Multidisciplinary System Development". In: *AIAA SCITECH 2023 Forum, January 2023, National Harbor (USA)*. 2023, p. 1895. DOI: 10.2514/6.2023-1895.

[66] OMG: *SysML v2: The Next Generation Systems Modeling Language!* 2025. URL: https://www.omg.org/sysml/sysmlv2/ (visited on 09/21/2025).

[67] OMG: *Object Management Group Approves Final Adoption of the SysML v2 Specification*. URL: https://www.omg.org/news/releases/pr2025/07-21-25.htm (visited on 07/21/2025).

[68] Molnár, V., Graics, B., Vörös, A., et al.: "Towards the Formal Verification of SysML v2 Models". In: *Proceedings of the ACM/IEEE 27th International Conference on Model Driven Engineering Languages and Systems, September 2024, Linz (Austria)*. 2024, pp. 1086–1095. DOI: 10.1145/3652620.3687820.

[69] DeHart, J. K.: "Leveraging Large Language Models for Direct Interaction with SysML v2". In: *INCOSE International Symposium, July 2024, Dublin (Ireland)*. Vol. 34. 1. 2024, pp. 2168–2185. DOI: 10.1002/iis2.13262.

[70] Weilkiens, T.: *Systems Engineering mit SysML/UML: Anforderungen, Analyse, Architektur*. 3rd ed. Heidelberg: dpunkt.verl., 2014. ISBN: 978-3-86490-091-4.

[71] Chami, M. and Bruel, J.-M.: "A Survey on MBSE Adoption Challenges". In: *INCOSE EMEA Sector Systems Engineering Conference (INCOSE EMEASEC 2018), November 2018, Berlin (Germany)*. 2018. URL: https://hal.science/hal-02124402/ (visited on 07/09/2025).

[72] Cloutier, R.: "Current Modeling Trends in Systems Engineering". In: *INSIGHT* 18(2) (2015). Ed. by Miller, W., pp. 10–13. DOI: 10.1002/inst.12013.

[73] Flink, J., Mischke, R., Schaffert, K., et al.: "Orchestrating tool chains for model-based systems engineering with rce". In: *2022 IEEE Aerospace Conference (AERO), March 2022, Big Sky (USA)*. IEEE. 2022, pp. 1–9. DOI: 10.1109/AERO53065.2022.9843838.

[74] Chami, M., Aleksandraviciene, A., Morkevicius, A., et al.: "Towards Solving MBSE Adoption Challenges: The D3 MBSE Adoption Toolbox". In: *INCOSE International Symposium, July 2018, Washington DC (USA)* 28(1) (2018), pp. 1463–1477. ISSN: 2334-5837. DOI: 10.1002/j.2334-5837.2018.00561.x.

[75] Förster, F., Koldewey, C., Bernijazov, R., et al.: “Navigating viewpoints in MBSE: challenges, potential and pathways for stakeholder participation in industry”. In: vol. 5. 2025, pp. 2531–2540. DOI: 10.1017/pds.2025.10267.

[76] Huldt, T. and Stenius, I.: “State-of-practice survey of model-based systems engineering”. In: *Systems Engineering* 22(2) (2018), pp. 134–145. DOI: 10.1002/sys.21466.

[77] Malone, R., Friedland, B., Herrold, J., et al.: “Insights from Large Scale Model Based Systems Engineering at Boeing”. In: *INCOSE International Symposium, July 2016, Edinburgh (Scotland)*. Vol. 26. 1. 2016, pp. 542–555. DOI: 10.1002/j.2334-5837.2016.00177.x.

[78] Zhang, L., Chen, Z., Laili, Y., et al.: “MBSE 2.0: Toward more integrated, comprehensive, and intelligent MBSE”. In: *Systems* 13(7) (2025), p. 584. DOI: 10.3390/systems13070584.

[79] Doering, R.: “Rechnergestützter Entwurf lokaler elektrischer Energiesysteme zur Individualisierung der Kabinen-Energieversorgung von Verkehrsflugzeugen”. PhD thesis. Hamburg: Technische Universität Hamburg, 2021.

[80] Langermann, R.: “Beitrag zur durchgängigen Simulationsunterstützung im Entwicklungsprozess von Flugzeugsystemen”. PhD thesis. Braunschweig: Technische Universität Braunschweig, 2008.

[81] Mohr, B. A.: “Methodik für den modellbasierten Vorentwurf von Flugzeugsystemen”. PhD thesis. München: Technische Universität München, 2012.

[82] Balicki, W., Gowacki, P., and Loroch, L.: “Large aircraft reliability study as important aspect of the aircraft systems’ design changes and improvements”. In: *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering* 235(1) (2021), pp. 138–147. DOI: 10.1177/0954410020925626.

[83] Bornholdt, R.: “Systemübergreifende Analyse und Bewertung von Architekturvarianten neuartiger Flugzeugsysteme anhand von Sicherheits- und Betriebsaspekten”. PhD thesis. Hamburg: Technische Universität Hamburg, 2021.

[84] Howe, D.: *Aircraft Conceptual Design Synthesis*. Cranfield: John Wiley & Sons Ltd., 2000. DOI: 10.1002/9781118903094.

[85] Roemelt, S.: “Electrical Systems Engineering & Integration in AIRBUS”. In: *ICAS Biennial Workshop, August 2015, Krakow (Poland)*. 2015. URL: https://www.icas.org/media/pdf/Workshops/2015/ICAS_Workshop_Roemelt.pdf (visited on 07/09/2025).

[86] SAE International Recommended Practice: *Guidelines for Conducting the Safety Assessment Process on Civil Aircraft, Systems, and Equipment*. SAE Standard ARP4761A. Revised December 2023, Issued December 1996. DOI: https://doi.org/10.4271/ARP4761A.

[87] European Union Aviation Safety Agency: *Certification Specifications and Acceptable Means of Compliance for Large Aeroplanes (CS-25)*. 2021. URL: https://www.easa.europa.eu/en/document-library/easy-access-rules/easy-access-rules-large-aeroplanes-cs-25.

[88] Jeffroy, A., Nogué, J.-M., Cambet, M., et al.: "Airbus System Engineering Policy". In: *INCOSE International Symposium, June 1999, Brighton (England)*. Vol. 9. 1. 1999, pp. 291–296. DOI: 10.1002/j.2334-5837.1999.tb00173.x.

[89] Rosenow, H.: "Trade Off Bewertungsmethodik für Tool- und Methodenentscheidungen zur Virtualisierung und Modellbasierung in der Entwicklung". Master thesis. München: Technische Universität München, 2018.

[90] Stryker, C. and Kavlakoglu, E.: *What is AI?* IBM. 2024. URL: https://www.ibm.com/think/topics/artificial-intelligence?mhsrc=ibmsearch_a&mhq=what%20is%20AI (visited on 01/17/2025).

[91] International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC): *Information technology - Artificial intelligence - Artificial intelligence concepts and terminology*. ISO/IEC 22989:2022. 2022. URL: https://www.iso.org/obp/ui/en/#iso:std:iso-iec:22989:ed-1:v1:en (visited on 06/01/2025).

[92] Mohri, M., Rostamizadeh, A., and Talwalkar, A.: *Foundations of Machine Learning*. 2nd ed. Adaptive computation and machine learning. The MIT Press, 2018. ISBN: 9780262039406.

[93] Sarker, I. H.: "Machine learning: Algorithms, real-world applications and research directions". In: *SN computer science* 2(3) (2021), p. 160. DOI: 10.1007/s42979-021-00592-x.

[94] LeCun, Y., Bengio, Y., and Hinton, G.: "Deep learning". In: *Nature* 521(7553) (2015), pp. 436–444. DOI: 10.1038/nature14539.

[95] Caballar, R. D.: *What are foundation models?* IBM. 2024. URL: https://www.ibm.com/think/topics/foundation-models?utm_source=chatgpt.com (visited on 06/18/2025).

[96] Munro, K., Papp, S., Toth, Z., et al.: *The Handbook of Data Science and AI: Generate Value from Data with Machine Learning and Data Analytics*. 2nd edition. Hanser Publications, 2024. DOI: 10.3139/9781569902356.

[97] Thompson, A. D.: *Models Table*. 2025. URL: https://lifearchitect.ai/models-table/ (visited on 08/29/2025).

[98] Zhang, Z., Zhang, J., Zhang, X., et al.: "A comprehensive overview of Generative AI (GAI): Technologies, applications, and challenges". In: *Neurocomputing* (2025), p. 129645. DOI: 10.1016/j.neucom.2025.129645.

[99] Jin, K., Wang, Y., Santos, L., et al.: "Reasoning or Not? A Comprehensive Evaluation of Reasoning LLMs for Dialogue Summarization". In: *arXiv preprint arXiv:2507.02145* (2025). DOI: 10.48550/arXiv.2507.02145.

[100] Ye, Y., Fu, B., Huang, C., et al.: "How Well Do Large Reasoning Models Translate? A Comprehensive Evaluation for Multi-Domain Machine Translation". In: *arXiv preprint arXiv:2505.19987* (2025). DOI: 10.48550/arXiv.2505.19987.

[101] Hashemi, M., Bamgbose, O., Madhusudhan, S. T., et al.: "DNR Bench: Benchmarking Over-Reasoning in Reasoning LLMs". In: *arXiv preprint arXiv:2503.15793* (2025). DOI: 10.48550/arXiv.2503.15793.

[102] Jurafsky, D. and Martin, J. H.: *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models*. 3rd edition. New York: Stanford University, 2025. URL: https://web.stanford.edu/~jurafsky/slp3/ed3book_aug25.pdf (visited on 09/01/2025).

[103] Nie, T., Sun, J., and Ma, W.: "Exploring the roles of large language models in reshaping transportation systems: A survey, framework, and roadmap". In: *Artificial Intelligence for Transportation* 1 (2025). DOI: 10.1016/j.ait.2025.100003.

[104] Sennrich, R., Haddow, B., and Birch, A.: "Neural machine translation of rare words with subword units". In: *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: long papers)* (2016), pp. 1715–1725.

[105] Wang, Z., Duan, F., Zhang, Y., et al.: "PositionID: Llms can Control Lengths, Copy and Paste with Explicit Positional Awareness". In: *arXiv preprint arXiv:2410.07035* (2024). DOI: 10.48550/arXiv.2410.07035.

[106] Gessner, R. R.: "AI Connections: Introduction to AI, methods, tools, and applications". Script. San Diego: University of California, San Diego, 2024.

[107] Vaswani, A., Shazeer, N., Parmar, N., et al.: "Attention Is All You Need". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems, December 2017, Long Beach (USA)*. Red Hook, NY, USA: Curran Associates Inc, 2017, pp. 6000–6010. DOI: 10.48550/arXiv.1706.03762.

[108] Bommasani, R., Hudson, D. A., Adeli, E., et al.: "On the opportunities and risks of foundation models". In: *arXiv preprint arXiv:2108.07258* (2021).

[109] Ouyang, L., Wu, J., Jiang, X., et al.: "Training language models to follow instructions with human feedback". In: *Advances in Neural Information Processing Systems 35 (NeurIPS 2022), December 2022, New Orleans (USA)*. Ed. by Koyejo, S., Mohamed, S., Agarwal, A., et al. Vol. 35. Curran Associates, Inc., 2022, pp. 27730–27744. URL: https://proceedings.neurips.cc/paper_files/paper/2022/file/b1efde53be364a73914f58805a001731-Paper-Conference.pdf.

[110] Koch, D., Kohne, A., and Brechbühler, N.: *Prompt Engineering in the Enterprise–An Introduction*. Springer Fachmedien Wiesbaden GmbH, 2025. DOI: 10.1007/978-3-658-49218-2.

[111] Schulhoff, S., Ilie, M., Balepur, N., et al.: "The Prompt Report: A Systematic Survey of Prompt Engineering Techniques". In: *arXiv preprint arXiv:2406.06608* (2025). DOI: 10.48550/arXiv.2406.06608.

[112] Kistowski, J. v., Arnold, J. A., Huppler, K., et al.: "How to Build a Benchmark". In: *Proceedings of the 6th ACM/SPEC International Conference on Performance Engineering, February 2015, Austin (USA)*. New York, NY, USA: Association for Computing Machinery, 2015, pp. 333–336. DOI: 10.1145/2668930.2688819.

[113] Caballar, R. D. and Stryker, C.: *What are LLM benchmarks?* IBM. 2024. URL: https://www.ibm.com/think/topics/llm-benchmarks (visited on 09/01/2025).

[114] DocsBot AI: *Compare o1 vs GPT-4.1*. 2025. URL: https://docsbot.ai/models/compare/o1/gpt-4-1#benchmarks (visited on 09/01/2025).

[115] Hendrycks, D., Burns, C., Basart, S., et al.: "Measuring Massive Multitask Language Understanding". In: *The 9th International Conference on Learning Representations (ICLR 2021), online*. 2020. DOI: 10.48550/arXiv.2009.03300.

[116] Rein, D., Hou, B. L., Stickland, A. C., et al.: "GPQA: A Graduate-Level Google-Proof Q&A Benchmark". In: *First Conference on Language Modeling*. 2024. DOI: 10.48550/arXiv.2311.12022.

[117] Chen, M., Tworek, J., Jun, H., et al.: "Evaluating large language models trained on code". In: *arXiv preprint arXiv:2107.03374* (2021). DOI: 10.48550/arXiv.2107.03374.

[118] Yue, X., Ni, Y., Zhang, K., et al.: "MMMU: A Massive Multi-discipline Multimodal Understanding and Reasoning Benchmark for Expert AGI". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, June 2024, Seattle (USA)*. 2024, pp. 9556–9567. DOI: 10.48550/arXiv.2311.16502.

[119] Zhang, T., Ladhak, F., Durmus, E., et al.: "Benchmarking large language models for news summarization". In: *Transactions of the Association for Computational Linguistics* 12 (2024), pp. 39–57. DOI: 10.1162/tacl_a_00632.

[120] Hu, T. and Zhou, X.-H.: "Unveiling LLM Evaluation Focused on Metrics: Challenges and Solutions". In: *arXiv preprint arXiv:2404.09135* (2024). DOI: 10.48550/arXiv.2404.09135.

[121] Mustapha, K.: "A survey of emerging applications of large language models for problems in mechanics, product design, and manufacturing". In: *Advanced Engineering Informatics* 64 (2024), p. 103066. DOI: 10.1016/j.aei.2024.103066.

[122] Zadenoori, M. A., Dbrowski, J., Alhoshan, W., et al.: "Large Language Models (LLMs) for Requirements Engineering (RE): A Systematic Literature Review". In: *arXiv preprint arXiv:2509.11446v1* (2025). Preprint submitted to Elsevier. DOI: 10.48550/arXiv.2509.11446.

[123] Chiarello, F., Barandoni, S., Skec, M. M., et al.: "Generative large language models in engineering design: opportunities and challenges". In: *Proceedings of the Design Society, May 2024, Cavtat (Croatia)*. Vol. 4. Cambridge University Press, 2024, pp. 1959–1968. DOI: 10.1017/pds.2024.198.

[124] Raza, M., Jahangir, Z., Riaz, M. B., et al.: "Industrial applications of large language models". In: *Scientific Reports* 15(1) (2025), p. 13755. DOI: 10.1038/s41598-025-98483-1.

[125] Schmid, L., Hey, T., Armbruster, M., et al.: "Software Architecture Meets LLMs: A Systematic Literature Review". In: *arXiv preprint arXiv:2505.16697* (2025). DOI: 10.48550/arXiv.2505.16697.

[126] Zhang, Y., Khan, S. A., Mahmud, A., et al.: "Exploring the role of large language models in the scientific method: from hypothesis to discovery". In: *npj Artificial Intelligence* 1(1) (2025), p. 14. DOI: 10.1038/s44387-025-00019-5.

[127] Cheng, J., Marone, M., Weller, O., et al.: "Dated Data: Tracing Knowledge Cutoffs in Large Language Models". In: *First Conference on Language Modeling, October 2024, Philadelphia (USA)*. 2024. URL: https://openreview.net/forum?id=wS7PxDjy6m.

[128] Zhang, Y., Li, Y., Cui, L., et al.: “Siren’s Song in the AI Ocean: A Survey on Hallucination in Large Language Models”. In: *Computational Linguistics* (2023), pp. 1–45. DOI: 10.1162/coli.a.16.

[129] Mirzadeh, I., Alizadeh, K., Shahrokhi, H., et al.: “GSM-Symbolic: Understanding the Limitations of Mathematical Reasoning in Large Language Models”. In: *ICLR Proceedings of the 13th International Conference on Learning Representations, April 2025, Singapore*. 2024. URL: https://openreview.net/pdf?id=AjXkRZIvjB (visited on 07/09/2025).

[130] Klesel, M. and Wittmann, H. F.: “Retrieval-Augmented Generation (RAG)”. In: *Business & Information Systems Engineering* 67 (2025), pp. 551–561. DOI: 10.1007/s12599-025-00945-3.

[131] Gao, Y., Xiong, Y., Gao, X., et al.: “Retrieval-Augmented Generation for Large Language Models: A Survey”. In: *arXiv preprint arXiv:2312.10997* (2024). Preprint. DOI: 10.48550/arXiv.2312.10997.

[132] Lewis, P., Perez, E., Piktus, A., et al.: “Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks”. In: *Proceedings of the 34th Conference on Neural Information Processing Systems (NeurIPS), December 2020, Vancouver (Canada)*. 2020. DOI: 10.48550/arXiv.2005.11401.

[133] Ben, L.: *Best Practices in Retrieval Augmented Generation*. Gradient Flow. Oct. 19, 2023. URL: https://gradientflow.com/best-practices-in-retrieval-augmented-generation/ (visited on 07/09/2025).

[134] Gan, A., Yu, H., Zhang, K., et al.: “Retrieval Augmented Generation Evaluation in the Era of Large Language Models: A Comprehensive Survey”. In: *arXiv preprint arXiv:2504.14891* (2025). DOI: 10.48550/arXiv.2504.14891.

[135] Honroth, T., Siebert, J., and Kelbert, P.: *Retrieval Augmented Generation (RAG): Chatten mit den eigenen Daten*. Fraunhofer. May 30, 2025. URL: https://www.iese.fraunhofer.de/blog/retrieval-augmented-generation-rag/ (visited on 05/13/2024).

[136] Yu, H., Gan, A., Zhang, K., et al.: “Evaluation of Retrieval-Augmented Generation: A Survey”. In: *CCF Conference on Big Data, August 2024, Qingdao (China)*. Springer Nature Singapore. 2024, pp. 102–120. DOI: 10.1007/978-981-96-1024-2_8.

[137] Ganesan, K.: “Rouge 2.0: Updated and Improved Measures for Evaluation of Summarization Tasks”. In: *arXiv preprint arXiv:1803.01937* (2018). DOI: 10.48550/arXiv.1803.01937.

[138] Zheng, L., Chiang, W.-L., Sheng, Y., et al.: “Judging llm-as-a-judge with mt-bench and chatbot arena”. In: *Advances in Neural Information Processing Systems 36 (NeurIPS 2023), December 2023, New Orleans (USA)*. Vol. 36. 2023, pp. 46595–46623. DOI: 10.48550/arXiv.2306.05685.

[139] Dietz, L., Zendel, O., Bailey, P., et al.: “Principles and Guidelines for the Use of LLM Judges”. In: *Proceedings of the 2025 International ACM SIGIR Conference on Innovative Concepts and Theories in Information Retrieval (ICTIR), July 2025, Padua (Italy)*. New York, NY, USA: Association for Computing Machinery, 2025, pp. 218–229. DOI: 10.1145/3731120.3744588.

[140] Fu, J., Qin, X., Yang, F., et al.: "AutoRAG-HP: Automatic Online Hyper-Parameter Tuning for Retrieval-Augmented Generation". In: *Findings of the Association for Computational Linguistics: EMNLP 2024, November 2024, Miami (USA)*. Association for Computational Linguistics, 2024, pp. 3437–3451. DOI: 10.48550/arXiv.2406.19251. URL: https://aclanthology.org/2024.findings-emnlp.223.

[141] Weber, R.: "On the Gittins index for multiarmed bandits". In: *The Annals of Applied Probability* 2(4) (1992), pp. 1024–1033. DOI: 10.1214/aoap/1177005588.

[142] Ammar, A., Koubaa, A., Nacar, O., et al.: "Optimizing Retrieval-Augmented Generation: Analysis of Hyperparameter Impact on Performance and Efficiency". In: *arXiv preprint arXiv:2505.08445* (2025). Preprint. DOI: 10.48550/arXiv.2505.08445.

[143] Barker, M., Bell, A., Thomas, E., et al.: "Faster, Cheaper, Better: Multi-Objective Hyperparameter Optimization for LLM and RAG Systems". In: *arXiv preprint arXiv:2502.18635* (2025). Preprint. DOI: 10.48550/arXiv.2502.18635.

[144] Feldman, P., Foulds, J. R., and Pan, S.: "RAGged Edges: The Double-Edged Sword of Retrieval-Augmented Chatbots". In: *arXiv preprint arXiv:2403.01193* (2024). DOI: 10.48550/arXiv.2403.01193.

[145] Ji, Z., Lee, N., Frieske, R., et al.: "Survey of Hallucination in Natural Language Generation". In: *ACM Computing Surveys* 55(12) (2022), pp. 1–38. DOI: 10.1145/3571730.

[146] Xie, X., Tang, X., Gu, S., et al.: "An intelligent guided troubleshooting method for aircraft based on HybirdRAG". In: *Scientific Reports* 15(1) (2025), p. 17752. DOI: 10.1038/s41598-025-02643-2.

[147] Zheng, T. et al.: "A Retrieval-Augmented Generation Method for QA on Airworthiness Regulations". In: *Electronics* (2025). DOI: 10.3390/electronics14163314.

[148] Xiong, X., Cai, H., Yu, H., et al.: "DR-RAG: Domain-Rule-based Retrieval-Augmented Generation for aviation digital model design". In: *Advanced Engineering Informatics* 68 (2025), p. 103688. DOI: 10.1016/j.aei.2025.103688.

[149] deepset: *Case Study: Airbus*. 2024. URL: https://www.deepset.ai/case-studies/airbus.

[150] Press, O., Zhang, M., Min, S., et al.: "Measuring and narrowing the compositionality gap in language models". In: *arXiv preprint arXiv:2210.03350* (2022). DOI: 10.48550/arXiv.2210.03350.

[151] Trivedi, H., Balasubramanian, N., Khot, T., et al.: "Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions". In: *arXiv preprint arXiv:2212.10509* (2022). DOI: 10.48550/arXiv.2212.10509.

[152] Wang, L., Chen, H., Yang, N., et al.: "Chain-of-Retrieval Augmented Generation". In: *arXiv preprint arXiv:2501.14342v2* (2025). DOI: 10.48550/arXiv.2501.14342.

[153] Shao, Z., Gong, Y., Shen, Y., et al.: "Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy". In: *arXiv preprint arXiv:2305.15294* (2023). DOI: 10.48550/arXiv.2305.15294.

[154] Jiang, Z., Xu, F. F., Gao, L., et al.: “Active retrieval augmented generation”. In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, December 2023, Singapore*. 2023, pp. 7969–7992. DOI: 10.48550/arXiv.2305.06983.

[155] Asai, A., Wu, Z., Wang, Y., et al.: “Self-RAG: Learning to retrieve, generate, and critique through self-reflection”. In: *Proceedings of the International Conference on Learning Representations 2024, May 2024, Vienna (Austria)*. 2024. DOI: 10.48550/arXiv.2310.11511.

[156] Su, W., Tang, Y., Ai, Q., et al.: “DRAGIN: dynamic retrieval augmented generation based on the information needs of large language models”. In: *arXiv preprint arXiv:2403.10081* (2024). DOI: 10.48550/arXiv.2403.10081.

[157] Sarthi, P., Abdullah, S., Tuli, A., et al.: “Raptor: Recursive abstractive processing for tree-organized retrieval”. In: *The 12th International Conference on Learning Representations, May 2024, Vienna (Austria)*. 2024. DOI: 10.48550/arXiv.2401.18059.

[158] Microsoft: *Welcome to GraphRAG*. 2024. URL: https://microsoft.github.io/graphrag/ (visited on 09/01/2025).

[159] Yan, S.-Q., Gu, J.-C., Zhu, Y., et al.: “Corrective retrieval augmented generation”. In: *ICLR 2025 Conference Withdrawn Submission*. 2024. DOI: 10.48550/arXiv.2401.15884.

[160] Ren, T., Zhang, Z., Jia, B., et al.: “Retrieval-Augmented Generation-aided causal identification of aviation accidents: A large language model methodology”. In: *Expert Systems with Applications* 278 (2025). DOI: 10.1016/j.eswa.2025.127306.

[161] Luo, Y., Xu, W., Zeng, C., et al.: “Application and evaluation of RAG technology in civil aviation policy question answering”. In: *2024 2nd International Conference on Computer Network Technology and Electronic and Information Engineering (CNTEIE), December 2024, Changchun (China)*. 2024. DOI: 10.1109/CNTEIE66268.2024.00032.

[162] Xiang, Z., Luo, Y., Fu, Q., et al.: “Civil Aviation Legal Retrieval and Analysis Based on RAG”. In: *Proceedings of the 5th International Conference on Artificial Intelligence and Computer Engineering, November 2024, Wuhu Shandong (China)*. 2024, pp. 351–357. DOI: 10.1145/3716895.3716958.

[163] Siddharth, L. and Luo, J.: “Retrieval augmented generation using engineering design knowledge”. In: *Knowledge-Based Systems* 303 (2024). DOI: 10.1016/j.knosys.2024.112410.

[164] Schlichting, M. R., Rasmussen, V., Alazzeh, H., et al.: “LeRAAT: LLM-Enabled Real-Time Aviation Advisory Tool”. In: *arXiv preprint arXiv:2503.16477* (2025). Preprint. DOI: 10.48550/arXiv.2503.16477.

[165] Pandey, S., Xu, R., Wang, W., et al.: “OpenFOAMGPT: a RAG-Augmented LLM Agent for OpenFOAM-Based Computational Fluid Dynamics”. In: *Physics of Fluids* 37(3) (2025). DOI: 10.1063/5.0257555.

[166] Silva, E. A., Marsh, R., Yong, H., et al.: "Retrieval-Augmented Generation and In-Context Prompted Large Language Models in Aircraft Engineering". In: *AIAA SCITECH 2025 Forum, January 2025, Orlando (USA)*. 2025. DOI: 10.2514/6.2025-0700.

[167] Zhang, Z., Yang, X., and Zeng, C.: "A Domain-specific Retrieval-Augmented Generation System for Civil Aviation Safety". In: *2024 4th International Conference on Big Data, Artificial Intelligence and Risk Management (ICBAR 2024), June 2024, Chengdu (China)*. 2024, pp. 682–687. DOI: 10.1145/3718751.3718860.

[168] Alarcia, R. M. G., Russo, P., Renga, A., et al.: "Bringing Systems Engineering Models to Large Language Models: An Integration of OPM with an LLM for Design Assistants." In: *Proceedings of the 12th International Conference on Model-Based Software and Systems Engineering - Volume 1: MBSE-AI Integration, February 2024, Rome (Italy)*. Ed. by Mayo, F., Pires, L., and Seidewitz, E. SciTePress, 2024, pp. 334–345. DOI: 10.5220/0012621900003645.

[169] Timperley, L. R., Berthoud, L., Snider, C., et al.: "Assessment of large language models for use in generative design of model based spacecraft system architectures". In: *Journal of Engineering Design* 36(4) (2025), pp. 550–570. DOI: 10.1080/09544828.2025.2453401.

[170] Timperley, L. and Berthoud, L.: "Reliability analysis and failure mitigation strategies for the PROVE pathfinder CubeSat payload". In: *2022 IEEE Aerospace Conference (AERO), March 2022, Big Sky (USA)*. IEEE. 2022, pp. 1–18. DOI: 10.1109/AERO53065.2022.9843234.

[171] Gomez, A. P., Krus, P., Panarotto, M., et al.: "Large language models in complex system design". In: *Proceedings of the Design Society, May 2024, Cavtat (Croatia)* 4 (2024), pp. 2197–2206. DOI: 10.1017/pds.2024.222.

[172] EASA: *AI Concept Paper Issue 2: Guidance for Level 1 & 2 ML Applications*. 2024. URL: https://www.easa.europa.eu/en/document-library/general-publications/easa-artificial-intelligence-concept-paper-issue-2.

[173] Lee, S. and Kim, M. S.: "Stack cooling system coupled with secondary heat pump in fuel cell electric vehicles". In: *Energy Conversion and Management* 284 (2023), p. 116961. DOI: 10.1016/j.enconman.2023.116961.

[174] Moxon, M.: "Thermal Management System For A Battery Powered Aircraft". European Patent Application EP 3 480 114 A1. European Patent Office (EPO). Oct. 2018. URL: https://patents.google.com/patent/EP3480114A1.

[175] Tiwari, R.: *Unlocking the Power of Sentence Embeddings with all-MiniLM-L6-v2*. Medium. 2024. URL: https://medium.com/@rahultiwari065/unlocking-the-power-of-sentence-embeddings-with-all-minilm-l6-v2-7d6589a5f0aa (visited on 06/01/2025).

[176] Wang, J.: *Embedding models: State-of-the-Arts*. Medium. 2025. URL: https://medium.com/@j.wang.mlds_97641/embedding-models-state-of-the-arts-b92a6872a7a1 (visited on 05/27/2025).

[177] Trabelsi, B.: *GPT From scratch for synthetic product reviews generation and market research*. Medium. Apr. 3, 2024. URL: https://medium.com/p/3c7444ea00cf (visited on 04/20/2025).

[178] Alammar, J. and Grootendorst, M.: *Hands-on large language models: language understanding and generation*. O'Reilly Media, Inc., 2024. ISBN: 978-1-098-15096-9.

[179] Watson, M. D., Griffin, M., Farrington, P. A., et al.: "Building a Path to Elegant Design". In: *Proceedings of the International Annual Conference of the American Society for Engineering Management, October 2014, Virginia Beach (USA)*. 2014, p. 1. ISBN: 978-1-63439-989-0. URL: https://web.mit.edu/ideation/papers/2014-griffinEtal.pdf (visited on 07/09/2025).

[180] Computer Security Resource Center (NIST): *Glossary: Parameter*. 2025. URL: https://csrc.nist.gov/glossary/term/parameter (visited on 09/02/2025).

[181] Ali, M.: *The Top 7 Vector Databases in 2025*. 2025. URL: https://www.datacamp.com/blog/the-top-5-vector-databases (visited on 01/18/2025).

[182] Miller, G. A.: "The Magical Number Seven, Plus or Minus Two: Some Limits on our Capacity for Processing Information". In: *Psychological Review* 63 (1956), pp. 81–97. URL: https://labs.la.utexas.edu/gilden/files/2016/04/MagicNumberSeven-Miller1956.pdf (visited on 09/01/2025).

[183] Powell, D. and Lorenz, R.: "The Effect of team size on the performance of continuous improvement teams: is seven really the magic number?" In: *IFIP International Conference on Advances in Production Management Systems. 1-5 September 2019, Austin (USA)*. Springer International Publishing, 2019, pp. 69–76. DOI: 10.1007/978-3-030-30000-5_9.

[184] Glaser, C.: *Risiko im Management: 100 Fehler, Irrtümer, Verzerrungen und wie man sie vermeidet*. 1st edition. Springer Gabler Wiesbaden, 2019. DOI: 10.1007/978-3-658-25835-1.

[185] Chandler, P. and Sweller, J.: "Cognitive Load Theory and the Format of Instruction". In: *Cognition and Instruction* 8(4) (1991), pp. 293–332. DOI: 10.1207/s1532690xci0804_2.

[186] DocsBot AI: *Detailed comparison of AI language models*. 2025. URL: https://docsbot.ai/models/compare/gpt-3-5-turbo/o1-mini (visited on 09/01/2025).

[187] Voth, V.: "Analyse der Entstehungsmechanismen von Ölnebel in Flugzeugkabinen". Bachelor thesis. Braunschweig: Technische Universität Braunschweig, 2018.

[188] Bräunling, W. J. G.: *Flugzeugtriebwerke: Grundlagen, Aero-Thermodynamik, ideale und reale Kreisprozesse, Thermische Turbomaschinen, Komponenten, Emissionen und Systeme*. Springer Berlin, 2015. DOI: 10.1007/978-3-642-34539-5.

[189] Bender, D.: "Integration of exergy analysis into model-based design and evaluation of aircraft environmental control systems". In: *Energy* 137 (2017), pp. 739–751. DOI: 10.1016/j.energy.2017.05.182.

[190] Lehle, W. and Sauterleute, A.: "Konzeption und Entwicklung von Flugzeugklimatisierungsanlagen". In: *Praxis-Seminar Luftfahrt*. 2006. URL: https://www.fzt.haw-hamburg.de/pers/Scholz/dglr/hh/text_2006_06_01_AirSystems.pdf (visited on 07/12/2025).

[191] Michaelis, S.: "Implementation of the requirements for the provision of clean air in crew and passenger compartments using the aircraft bleed air system". Master thesis. Cranfield: Cranfield University, 2016.

[192] Yang, H., Zhang, X., Wang, C., et al.: "Experimental and theoretical study on a novel energy-saving ECS for commercial airliners". In: *Applied Thermal Engineering* 127 (2017), pp. 1372–1381. DOI: 10.1016/j.applthermaleng.2017.08.043.

[193] Wright, S., Dixon-Hardy, D., and Heggs, P. J.: "Aircraft air conditioning heat exchangers and atmospheric fouling". In: *Thermal Science and Engineering Progress* 7 (2018), pp. 184–202. DOI: 10.1016/j.tsep.2018.06.007.

[194] Garry Kasparov: "The chess master and the computer". In: *The New York Review of Books*. Vol. 57. 2010, pp. 16–19.

[195] Schlabe, D.: "Modellbasierte Entwicklung von Energiemanagement-Methoden für Flugzeug-Energiesysteme". PhD thesis. Dresden: Technische Universität Dresden, 2015.

[196] Faleiro, L.: "Summary of the European power optimised aircraft (POA) project". In: *25th International Congress of the Aeronautical Sciences (ICAS), September 2006, Hamburg (Germany)* (2006), pp. 3–8.

[197] Vredenborg, E.: "Modellbasierter Entwurf von Kühlsystemen für Brennstoffzellen in Verkehrsflugzeugen". PhD thesis. Hamburg: Technische Universität Hamburg and Shaker Verlag GmbH, 2015.

[198] Rouss, V., Lesage, P., Bégot, S., et al.: "Mechanical behaviour of a fuel cell stack under vibrating conditions linked to aircraft applications part I: Experimental". In: *International Journal of Hydrogen Energy* 33(22) (2008), pp. 6755–6765. DOI: 10.1016/j.ijhydene.2008.08.032.

[199] Lüdders, H. P.: "Modellbasierter Entwurf und Bewertung von multifunktionalen Brennstoffzellensystemen auf Flugzeugebene". PhD thesis. Hamburg: Technische Universität Hamburg, 2014. ISBN: 978-3-8440-2857-7.

[200] Grymlas, J.: "Systemautomatisierung für den multifunktionalen Betrieb von Brennstoffzellen in Verkehrsflugzeugen". PhD thesis. Hamburg: Technische Universität Hamburg and Shaker Verlag GmbH, 2017.

[201] Kurzweil, P.: *Brennstoffzellentechnik: Grundlagen, Komponenten, Systeme, Anwendungen*. 2nd ed. Springer Vieweg Wiesbaden, 2013. DOI: 10.1007/978-3-658-00085-1.

[202] Ruge, M.: "Entwicklung eines flüssigkeitsgekühlten Polymer-Elektrolyt-Membran-Brennstoffzellenstapels mit einer Leistung von 6,5 kW". PhD thesis. Zürich: Eidgenössischen Technischen Hochschule Zürich, 2003.

[203] O'Hayre, R., Cha, S.-W., Colella, W. G., et al.: *Fuel Cell Fundamentals*. 3rd ed. John Wiley & Sons, 2016. DOI: 10.1002/9781119191766.

[204] Peters, R., ed.: *Brennstoffzellensysteme in der Luftfahrt*. 1st ed. Springer Vieweg Berlin, Heidelberg, 2015. DOI: 10.1007/978-3-662-46798-5.

[205] Siemer, M.: "Lokale Entropieproduktionsraten in der Polymerelektrolyt-Membran-Brennstoffzelle". PhD thesis. Helmut-Schmidt-Universität; Universität der Bundeswehr Hamburg, 2007. DOI: 10.24405/344.

[206] Haidl, R.: “Entwurf und Analyse von Zwei-Phasen-Kühlsystem-Architekturen für Brennstoffzellen in elektrifizierten Verkehrsflugzeugen”. Master thesis. Stuttgart: Universität Stuttgart, 2022.

[207] Lelong, J., Errazine, A., and Blangero, A.: “Agentic RAG with Knowledge Graphs for Complex Multi-Hop Reasoning in Real-World Applications”. In: *arXiv preprint arXiv:2507.16507* (2025). DOI: 10.48550/arXiv.2507.16507.

[208] Liang, J., Su, G., Lin, H., et al.: “Reasoning RAG via System 1 or System 2: A Survey on Reasoning Agentic Retrieval-Augmented Generation for Industry Challenges”. In: *arXiv preprint arXiv:2506.10408* (2025). Preprint.

[209] Li, Y., Zhang, W., Yang, Y., et al.: “Towards Agentic RAG with Deep Reasoning: A Survey of RAG-Reasoning Systems in LLMs”. In: *arXiv preprint arXiv:2507.09477* (2025). Preprint. DOI: 10.48550/arXiv.2507.09477.

[210] Murthy, D. and Page, N.: *Education and Research for the Future: Engineering as an Illustrative Case*. 1st. Springer Cham, 2023. DOI: 10.1007/978-3-031-29685-7.

[211] Stachowiak, H.: *Allgemeine Modelltheorie*. Wien, New York: Springer-Verlag, 1973. ISBN: 978-3-319-57444-8.

[212] Estefan, J. A.: “Survey of model-based systems engineering (MBSE) methodologies”. In: *Incose MBSE Focus Group* 25(8) (2008), pp. 1–12.

[213] Raksch, C.: “Eine Methode zur optimalen Redundanzallokation im Vorentwurf fehlertoleranter Flugzeugsysteme”. PhD thesis. Technische Universität Hamburg, 2013.

[214] Moir, I., Seabridge, A., and Jukes, M.: *Civil avionics systems*. 2nd ed. Aerospace series. John Wiley & Sons, 2013. ISBN: 978-1-118-34180-3.

[215] Turing, A.: “Computing Machinery And Intelligence”. In: *Mind* LIX(236) (1950), pp. 433–460. DOI: 10.1093/mind/LIX.236.433.

[216] McCarthy, J., Minsky, M. L., Rochester, N., et al.: “A Proposal for the Dartmouth Summer Research Project on Artificial Intelligence, August 31, 1955”. In: *AI Magazine* 27(4) (1955), pp. 12–14. DOI: 10.1609/aimag.v27i4.1904.

[217] Rosenblatt, F.: “The perceptron: a probabilistic model for information storage and organization in the brain.” In: *Psychological Review* 65(6) (1958), pp. 386–408. URL: https://homepages.math.uic.edu/~lreyzin/papers/rosenblatt58.pdf (visited on 07/09/2025).

[218] IBM: *Watson, ‘Jeopardy!’ champion*. URL: https://www.ibm.com/history/watson-jeopardy (visited on 06/01/2025).

[219] Alom, M. Z., Taha, T. M., Yakopcic, C., et al.: “The History Began from AlexNet: A Comprehensive Survey on Deep Learning Approaches”. In: *arXiv preprint arXiv:1803.01164* (2018). DOI: 10.48550/arXiv.1803.01164.

[220] Goodfellow, I.: “NIPS 2016 Tutorial: Generative Adversarial Networks”. In: *arXiv preprint arXiv:1701.00160* (2016). DOI: 10.48550/arXiv.1701.00160.

[221] OpenAI: *Improving language understanding with unsupervised learning*. 2018. URL: https://openai.com/index/language-unsupervised/ (visited on 06/01/2025).

[222] Gotel, O. and Finkelstein, C.: "An analysis of the requirements traceability problem". In: *Proceedings of IEEE International Conference on Requirements Engineering, April 1994, Colorado Springs (USA)*. 1994, pp. 94–101. DOI: 10.1109/ICRE.1994.292398.

[223] Kallo, J., Renouard-Vallet, G., Saballus, M., et al.: "Fuel Cell System Development and Testing for Aircraft Applications". In: *Proceedings of the 18th World Hydrogene Energy Conference (WHEC), May 2010, Essen*. 2010. ISBN: 978-3-89336-655-2.

[224] Deutsches Zentrum für Luft- und Raumfahrt (DLR): *Zero Emission Aviation - Emissionsfreie Luftfahrt*. 2020. URL: https://www.dlr.de/de/medien/publikationen/sonstige-publikationen/2020/white-paper-zero-emission-aviation (visited on 07/05/2024).

[225] Berger, O.: "Thermodynamische Analyse eines Brennstoffzellensystems zum Antrieb von Kraftfahrzeugen". PhD thesis. Duisburg-Essen: Universität Duisburg-Essen, 2009.

[226] Becker, F.: "Modellierung und Charakterisierung eines PEM-Brennstoffzellensystems mit partieller Rezirkulation des Kathodenabgases für den Einsatz im Flugzeug". Master thesis. Hamburg: Technische Universität Hamburg, 2015.

[227] Schmid, K.-H.: "Wärmemanagement von Brennstoffzellen-Elektrofahrzeugen: Zugl.: Stuttgart, Univ., Diss., 2009". PhD thesis. Stuttgart: Universität Stuttgart, 2009. ISBN: 9783867278898. URL: https://cuvillier.de/de/shop/publications/1183-warmemanagement-von-brennstoffzellen-elektrofahrzeugen (visited on 07/09/2025).

[228] Burschyk, T.: "Entwicklung und Bewertung eines brennstoffbasierten Klimatisierungssystems für luftfahrttechnische Anwendungen". Master thesis. Hamburg: Technische Universität Hamburg, 2020.

[229] Larminie, J., Dicks, A., and McDonald, M. S.: *Fuel Cell Systems Explained*. John Wiley & Sons Ltd. Ltd., 2003. DOI: 10.1002/9781118706992.

[230] Scheiderer, J.: *Angewandte Flugleistung: Eine Einführung in die operationelle Flugleistung vom Start bis zur Landung*. Springer Berlin, Heidelberg, 2008. DOI: 10.1007/978-3-540-72724-8.

[231] PowerCell: *PowerCellution P Stack*. URL: https://powercellgroup.com/wp-content/uploads/2022/05/p-stack-v-221.pdf (visited on 05/27/2024).

[232] Fly, A. and Thring, R.: "System thermal and water balance in an evaporatively cooled PEM fuel cell vehicle". In: *Vehicle Thermal Management Systems Conference Proceedings (VTMS11), May 2013, Coventry (UK)*. Woodhead Publishing, 2013, pp. 267–277. DOI: 10.1533/9780857094735.6.267.

A. Appendix

Appendix Numbering Convention

All elements in the appendix use prefixed numbering to distinguish them from the main document:

- Chapters: A.X
- Tables: AT.X (Appendix Table)
- Figures: AF.X (Appendix Figure)
- Equations: AE.X (Appendix Equation)

A.1. Definitions of Technical Terminology

This section defines the central technical and methodological terminology used throughout this dissertation. The consistent and unambiguous use of key terms is essential to ensure conceptual clarity, especially in interdisciplinary contexts where identical expressions may carry different meanings across domains. The following definitions consolidate established standards from SE and related disciplines, complemented by specific contextualizations that reflect their intended usage within this work. Each entry provides a concise definition and clarifies how the respective concept is applied in the context of the SAGE methodology and its case studies.

System

Definition: A system is defined as a combination of "*elements, subsystems, or assemblies that accomplish a defined objective. These elements include products (hardware, software, firmware), processes, people, information, techniques, facilities, services, and other support elements*" [12, p.265]. A system exhibits several fundamental characteristics. Its most distinctive property is the emergence of capabilities that individual components cannot provide on their own. These emergent properties result from the dynamic interplay and mutual dependence of system elements, which collectively achieve objectives beyond the scope of isolated parts. A system is further defined by clear boundaries that separate it from its environment, establishing the context in which it operates. Finally, every system is purpose-driven. It is conceived, designed, and implemented to fulfill specific functions or objectives within a defined operational domain.

Usage in this dissertation: The term *system* refers specifically to technical assemblies in aerospace applications, such as ECS or TMS, that integrate multiple elements to achieve defined operational objectives.

Systems Engineering (SE)

Definition: Systems Engineering is an interdisciplinary approach aimed at managing the increasing complicatedness of multidisciplinary technical systems and minimizing their risks. It encompasses systematic methodologies for structured development, through requirements documentation, functional analysis, and architectural modeling across all system levels [44, p.1], [49, p.3], [11].

Usage in this dissertation: Systems Engineering refers specifically to the systematic, interdisciplinary approach to technical system development, distinguished from individual engineering disciplines or ad-hoc development practices.

Engineering

Definition: The application of "*scientific principles to design or develop structures, machines, apparatus, or manufacturing processes, or works utilizing them singly or in combination; or to construct or operate the same with full cognizance of their design; or to forecast their behaviour under specific operating conditions; all as respects an intended function, economics of operation and safety to life and property*" [210, p.82].

Engineering encompasses both the systematic application of scientific knowledge and the creative problem-solving process required to develop practical solutions for real-world challenges. It is characterized by the integration of theoretical understanding with practical implementation, emphasizing functionality, safety, efficiency, and sustainability. Engineering differs from pure science in its focus on practical application and its consideration of multiple constraints, including cost, materials, time, and societal impact.

Key Characteristics are the application of scientific and mathematical principles, creative problem-solving and design thinking, consideration of practical constraints (economic, environmental, social), integration of theoretical knowledge with practical implementation, and the focus on solutions that benefit society.

Usage in this dissertation: Engineering can be understood as the structured use of scientific principles to address technical challenges in aerospace system development, integrating both analytical and creative problem-solving approaches.

Model

Definition: Following Eigner et al. [47, p.80] definition, a model is defined as "*a representation or template for a system or process*" that "*can be a conceptual (e.g., mathematical, information technology) or a physical (e.g., scaled material prototype) structure*" (author's translation from German). And according to Stachowiak [211], models are distinguished by three fundamental features: Mapping characteristic (every model serves as a representation or template for something else), reduction characteristic (every model abstracts from reality, focusing on relevant aspects while omitting others), pragmatic characteristic (every model is developed for a defined objective and intended application context).

Usage in this dissertation: A model encompasses both process models (structured development frameworks) and system models (representations of technical systems), with the specific type clarified by context.

Process Model

Definition: A specialized type of model that provides a structured framework defining the sequence of activities, phases, and decision points for organizing engineering development work to achieve specific objectives. Typical process models are i.e. the V-Model in the mechanical and aerospace engineering field or the Waterfall, Spiral or Agile Model in the field of software engineering [50, p.213],

Usage in this dissertation: Process model refers specifically to the V-Model and other systematic frameworks for organizing engineering development activities, distinguished from other model types such as mathematical or physical models.

System Model

Definition: A formal, structured representation of a technical system that captures different aspects of that system depending on the specific domain and purpose of the modeling activity.

System models serve as abstract representations that focus on relevant system characteristics for a given application context. The specific aspects captured vary by engineering discipline and modeling objectives: in systems engineering, these typically include requirements, functions, logical elements, and physical components. In software engineering, they may emphasize data structures, algorithms, and interfaces. System models enable systematic analysis, simulation, and communication throughout the development lifecycle while maintaining consistency across multiple system perspectives and abstraction levels.

System models in the SE field can be categorized into several types based on their focus and purpose. Structural models represent system architecture and component relationships, providing a hierarchical view of how system elements are organized and connected. Behavioral models describe system functions and operational sequences, detailing how the system responds to inputs and transitions between different states. Interface models define interactions between system elements and external entities, specifying the boundaries and communication protocols. Requirements models link system capabilities to stakeholder needs, ensuring traceability from user requirements to technical implementation. [12], [11]

Usage in this dissertation: System model refers to formal representations of technical systems (such as aircraft environmental control systems) that capture requirements, functions, logical elements, and physical components within structured frameworks like the RFLP approach.

Parameter

Definition: The word parameter refers to all variable configuration points within the methodology that can be adjusted to optimize system performance. These parameters represent the controllable aspects of the developed methodology that directly influence output quality and system behavior. In the sense of computer science, and following the NIST definition, a parameter is "*a value that is used to control the operation of a function or that is used by a function to compute one or more outputs*" [180]. This clarification is important, as the notion of parameter may differ in other disciplines, such as mechanical engineering, where it often denotes fixed material or design constants rather than configurable system inputs.

Usage in this dissertation: In the context of this dissertation, parameters define the tunable variables that influence the performance and output quality of the SAGE methodology across its different process stages. They include, for example, configuration values for vector-based retrievals, chunk sizes, embedding dimensions, model selection, as well as structural settings that guide the consolidation and evaluation stages. These parameters were systematically varied and empirically analyzed in Chapter 4 to identify best configurations for knowledge synthesis and architecture generation. In this sense, parameters serve as control variables that enable reproducible tuning and performance optimization of the overall methodology within the PhD project.

Methodology

Definition: According to Estefan [212, pp.2-3], a methodology is a structured collection of interrelated processes, methods, and tools that operate under coherent guiding principles. It provides the overarching rationale and theoretical foundation for systematic investigation and organized development work. A methodology thus defines the governing logic of how individual processes and methods interact to achieve consistent and reproducible results.

Usage in this dissertation: In this dissertation, the term *methodology* refers to the SAGE methodology, the systematic, generative framework developed to acquire, synthesize, and structure engineering knowledge by using LLMs. Rather than representing a conventional engineering lifecycle, SAGE defines a high-level procedure that guides data preparation, recursive knowledge extraction, knowledge synthesis, and architecture generation, ensuring traceable and reproducible system design outputs.

Method

Definition: A method is a concrete and repeatable procedure applied to accomplish a defined activity within a defined process. It operationalizes the *how* of an activity through structured steps, algorithms, or analytical techniques [212, p.2]. In SE, methods provide the practical means for implementing defined processes and for generating, transforming, or analyzing information in a controlled manner.

Usage in this dissertation: Within the SAGE methodology, methods represent the specific operational techniques used to realize each process stage. Examples include prompt engineering for guiding LLM outputs, semantic similarity search for knowledge retrieval, and recursive synthesis for consolidating distributed information. Each method defines an executable and verifiable procedure contributing to the overall reliability and traceability of the methodology.

Process

Definition: A process is defined according to Estefan [212, p.2] as a "*logical sequence of tasks performed to achieve a particular objective.*" It outlines *what* tasks are to be performed and in which order, establishing the logical and procedural flow that links methods and results. Processes serve as the organizing layer that connects methodological intent with concrete execution.

Usage in this dissertation: In the context of the SAGE methodology, processes define the ordered progression of activities that structure the development of AI-supported

system knowledge. The core processes comprise data preprocessing, recursive knowledge extraction, knowledge synthesis, and architecture generation. Each process provides the framework within which specific methods are applied, ensuring consistent and traceable knowledge transformation throughout the methodology.

Tool

Definition: A tool is a technical or software-based instrument that supports the execution of a method or process. Tools provide the practical means for implementing methodological activities efficiently and consistently. According to Estefan [212, p.3], tools form an integral part of a methodology, as they facilitate the application of processes and methods through automation, standardization, or integration of information flows.

Usage in this dissertation: Within the SAGE methodology, tools denote the software environments, scripts, and technical components that enable the operationalization of methods and processes.

Framework

Definition: A framework is a conceptual structure that organizes principles, categories, and relationships within a defined domain. It provides orientation and boundaries for the consistent integration of processes, methods, and tools, but does not describe their exact execution. Frameworks typically serve to ensure coherence and traceability across multiple abstraction levels and viewpoints in complex systems.

Usage in this dissertation: In this dissertation, the term *framework* refers to several conceptual structures that guide reasoning and organization within the SAGE methodology. The RFLP framework (Requirements-Functional-Logical-Physical) provides the technical backbone for structuring system architectures and ensuring traceability between requirements and physical implementations. The Cynefin framework offers a cognitive lens for classifying problem contexts by complexity and uncertainty, supporting methodological decisions on appropriate reasoning modes. The Elegant Design framework complements these by emphasizing balanced trade-offs and system simplicity as design principles.

Approach

Definition: An approach describes a general strategic orientation or guiding philosophy for addressing a problem. It is broader and less formalized than a methodology but more structured than an isolated method. An approach defines the overall mindset, priorities, and abstraction level under which methodologies and processes are developed and applied.

Usage in this dissertation: In this dissertation, the term *approach* refers to the overarching strategic perspective that integrates classical systems engineering principles with generative artificial intelligence. This approach provides the conceptual direction from which the SAGE methodology was derived, establishing the rationale for combining data-driven reasoning, language-model-based knowledge synthesis, and formal systems engineering structures.

A.2. ARP Guidelines for Aircraft System Development

Note: The following description is based on the author's prior publication [3] and has been slightly adapted.

In the development of safety-critical aircraft systems, the Certification Specifications for Large Airplanes (CS-25), as well as the FAA certification guidelines in the USA, refer to two aviation-specific standards: ARP4754B [28] and ARP4761A [86]. Together, they form the methodological foundation for system development and safety assessment. ARP4754B primarily addresses the structured development process of aircraft systems, including their order and recommended content. And the ARP4761A focuses on the safety assessment process and defines the analysis techniques applied within this framework. It is worth noting that both standards have recently been revised: ARP4754B ("*Guidelines for Development of Civil Aircraft and Systems*") [28] and ARP4761A ("*Guidelines for Conducting the Safety Assessment Process on Civil Aircraft, Systems, and Equipment*" [86] were most recently updated in 2023, continuing the evolution of these guidelines since their first publication in 1996 respectively. The interaction between both processes is depicted in Figure AF.1, which illustrates how system development activities (left) are continuously accompanied by safety assessment steps (right).

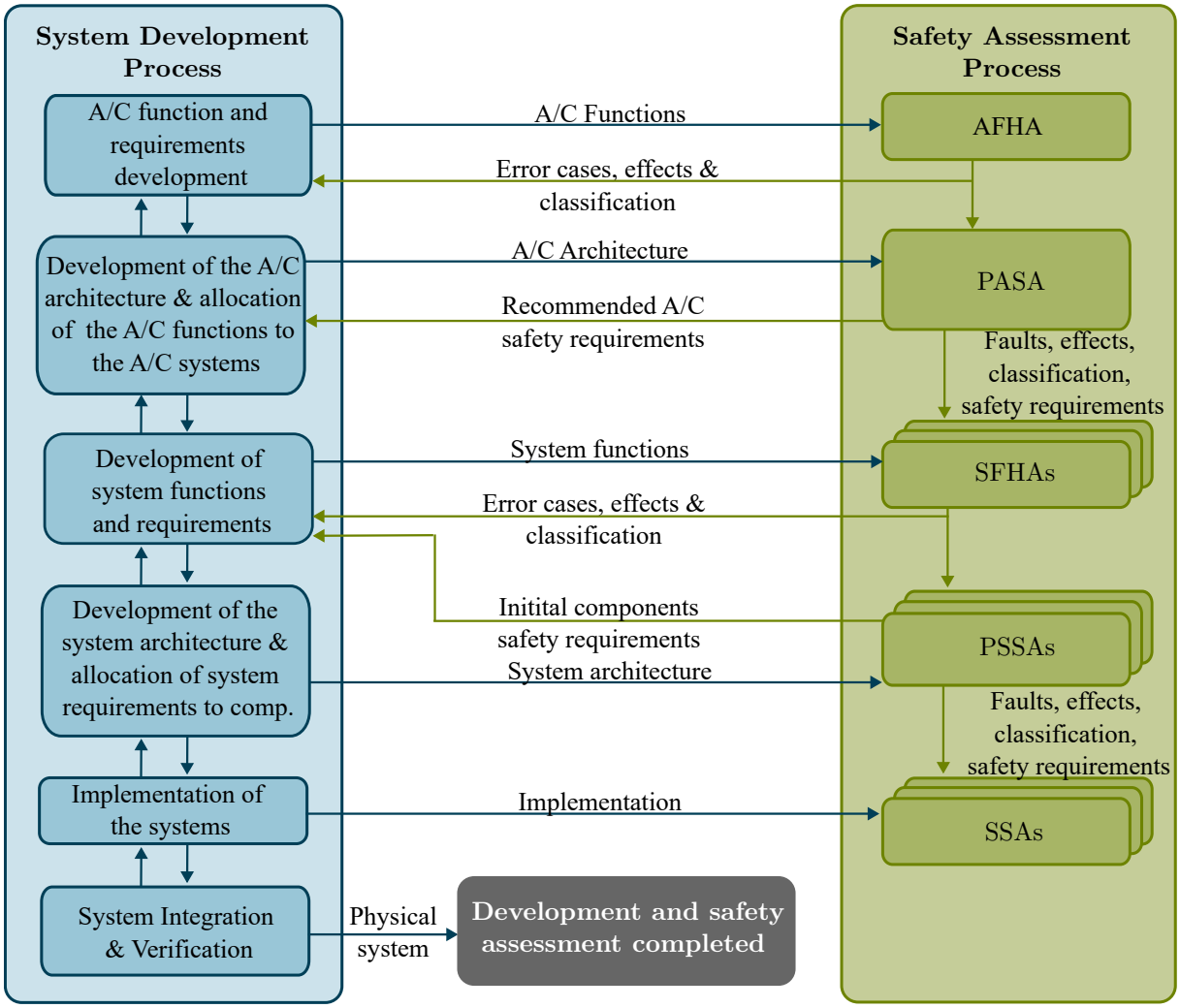

Figure AF.1.: Dependency between the system development and the safety assessment, adapted from [28, p.32]

The **system development** process begins with the identification of aircraft-level functions, which at this stage are not linked to any specific technical design [83]. As shown in Figure AF.1, these functions are then allocated to the individual aircraft systems, allowing the derivation of specific system functions. In the next step, a preliminary system architecture is defined based on these system functions. This stage plays a crucial role during the conceptual design phase, as all subsequent activities build upon the defined architecture [83]. Hardware and software components, derived from reliability and safety requirements, are subsequently integrated into the system architecture, and the overall system topology is defined. According to Raksch [213], topology describes the spatial arrangement and interconnection of all system components within the aircraft, taking into account installation space, wiring lengths, and integration constraints.

In parallel, the **safety assessment process** ensures that the developed system complies with CS-25 safety requirements. The first step at the aircraft level is the Aircraft Functional Hazard Analysis (A-FHA), which systematically examines aircraft-level functions to identify and classify potential hazards [28]. As indicated in Figure AF.1, the results of the A-FHA feed into the Preliminary Aircraft Safety Assessment (PASA). The PASA evaluates the assumed aircraft architecture for safety robustness by analyzing possible combinations of failure modes and their consequences. Methods typically applied in this context include Fault Tree Analysis (FTA) and Reliability Block Diagrams (RBD).

At the system level, the System Functional Hazard Analysis (S-FHA) identifies and classifies potential failures of each safety-critical system [213]. Figure AF.1 highlights how the S-FHA links system functions with possible error cases, effects, and safety requirements. Importantly, this analysis not only considers failures within a single system but also intersystem dependencies such as loss of power supply or data exchange. The next step, the Preliminary System Safety Assessment (PSSA), evaluates how the proposed system architecture (from the development process) reacts to failure conditions. As Figure AF.1 shows, the PSSA provides evidence that the system design fulfills both qualitative and quantitative safety requirements [86, 83, 214]. Furthermore, it identifies weaknesses in terms of safety and reliability, which can then be addressed by design improvements. Finally, the System Safety Assessment (SSA) represents the concluding step of the process. It systematically demonstrates that the implemented aircraft system complies with all safety objectives. In practice, this involves detailed analyses of failure modes, their effects, and compliance with regulatory safety criteria, ensuring that the aircraft system is safe for certification and operation.

A.3. Historical Development of AI

Early Foundations (1950-2000)

The foundations of AI were laid in the mid-20th century, when Turing [215] introduced the Turing Test as a benchmark for machine intelligence in his paper "Computing Machinery and Intelligence". This publication established the theoretical basis for what would later evolve into a transformative field of research. The formal beginning of AI as a distinct discipline occurred six years later, when John McCarthy introduced "Artificial Intelligence" at the Dartmouth Conference in 1956, establishing the academic and research framework that continues to guide the field today [216].

Significant technological advances followed throughout the subsequent decades. Frank Rosenblatt's development of the Mark 1 Perceptron in 1967 represented a crucial breakthrough in neural network learning capabilities, demonstrating that machines could learn from data through adaptive algorithms [217]. The field gained further momentum in 1986 when the backpropagation algorithm achieved widespread adoption, enabling more effective training of neural networks by allowing error signals to propagate backward through network layers. This period culminated in a landmark achievement in 1997, as IBM's Deep Blue prevailed over world chess champion Garry Kasparov, demonstrating AI's potential for complex strategic reasoning and marking the first time a computer system outperformed a human world champion in a game of such complexity [194].

This foundational period established key theoretical concepts and initial implementations, though generative capabilities remained limited by computational and algorithmic constraints. Engineering applications during this era primarily focused on expert systems for diagnostic and design support rather than content generation, reflecting the technology's emphasis on rule-based reasoning and knowledge representation.

Machine Learning Revolution (2000-2015)

The beginning of the 21st century represented a decisive phase in the evolution of AI, driven by the convergence of several key technological trends. The emergence of Big Data and cloud computing in the early 2000s provided the necessary computational infrastructure for advanced AI applications, allowing researchers to handle vast datasets and run sophisticated algorithms at scale. This infrastructure development proved essential for the subsequent machine learning breakthroughs.

A significant milestone occurred in 2011 when IBM's Watson system won the television quiz show "Jeopardy!", demonstrating the potential of machine learning for complex knowledge processing and natural language understanding [218]. Watson's ability to parse questions, access vast knowledge bases, and formulate responses showcased the practical applications of AI in knowledge-intensive domains. The following year brought perhaps the most transformative development of this period when AlexNet achieved breakthrough performance in the ImageNet competition in 2012, establishing deep neural networks as transformative technology and ushering in the deep learning revolution [219, p.11].

The introduction of Generative Adversarial Networks (GANs) by Goodfellow in 2014 represented a fundamental shift toward sophisticated content generation. This architecture established the theoretical and practical foundation for modern generative models by creating a competitive training framework between generator and discriminator networks [220]. GANs represented the first significant step toward sophisticated content generation with practical applications extending to technical domains including design optimization and anomaly detection in engineering systems.

This period was characterized by the practical application of deep learning techniques enabled by increased computational power and data availability. The combination of algorithmic advances, computational resources, and data abundance created the conditions necessary for the generative AI breakthroughs that would follow.

Transformer Revolution and GenAI Breakthrough (2017-Present)

The current era of AI development is defined by transformative advances in GenAI, beginning with the introduction of the Transformer architecture by Vaswani et al. [107] in 2017 through their influential paper "*Attention is All You Need.*" This architecture created the foundation for modern generative models by revolutionizing neural network design through the implementation of self-attention mechanisms as the central computational element, eliminating the need for recurrent structures that had previously limited scalability and parallelization.

The Transformer's impact became immediately apparent when OpenAI released the first Generative Pre-trained Transformer (GPT) model in 2018, demonstrating the architecture's potential for natural language generation [221]. This was followed by increasingly sophisticated implementations, culminating in GPT-3's release in 2020, which demonstrated unprecedented natural language capabilities through its 175 billion parameters and remarkable few-shot learning abilities. The scale and performance of GPT-3 represented a qualitative leap in AI capabilities, enabling applications that had long been regarded as unattainable for automated systems.

Simultaneously, the field recognized the limitations of purely generative approaches for knowledge-intensive tasks, leading Lewis et al. [132] to introduce RAG in 2020. This approach combined generative capabilities with external knowledge sources, addressing the hallucination problem inherent in LLMs while maintaining their generative flexibility. RAG systems demonstrated particular promise in domains requiring accurate, up-to-date information retrieval combined with sophisticated natural language generation.

he mainstream adoption of these technologies accelerated dramatically following the public introduction of ChatGPT in 2022, which marked the widespread integration of LLMs across diverse professional domains [34]. This release demonstrated the practical utility of conversational AI systems and catalyzed rapid adoption across industries. The period from 2023 onward has been characterized by the accelerated integration of retrieval mechanisms with generative models, particularly in knowledge-intensive domains where accuracy and verifiability are paramount.

The Transformer architecture introduced in 2017 represented a pivotal advancement, enabling scalable self-attention mechanisms that led to breakthroughs like GPT, and DALL-E [107]. This innovation revolutionized neural network architecture by employing attention mechanisms as the central element, eliminating the need for recurrent structures [107]. The full impact became evident with the emergence of LLMs like GPT-3, which demonstrated impressive few-shot learning capabilities [33, p.1].

For engineering applications, a significant development was the publication of the Retrieval-Augmented Generation (RAG) framework by Lewis et al. [132] in 2020, which addressed key limitations of generative models in knowledge-intensive domains. Through the integration of the generative capabilities of language models with retrieval mechanisms from authoritative sources, RAG has opened new opportunities for technical documentation generation, design support, and knowledge management in engineering domains such as aerospace. This approach has proven particularly valuable for domains requiring high precision, regulatory compliance, and traceability to authoritative sources.

A.4. Extraction and Analysis of Visual Document Elements

Although the main body of this dissertation focuses exclusively on textual content, early investigations explored the potential of integrating visual document elements such as diagrams, schematic architectures, and flow representations. This appendix documents the methodology, implementation steps, and results of these preliminary experiments. The objective of this work was to assess whether the combination of image extraction and surrounding text analysis could provide additional semantic insight into system architectures, particularly for ECS and TMS, where functional relationships are often described graphically.

Objective and Motivation

In technical-scientific literature, visual elements often carry a disproportionately high amount of information. Especially in system-related publications, architecture diagrams compress relationships into compact visual representations, frequently exceeding the informational density of accompanying text. However, these figures are challenging for automated interpretation because component names, interconnections, and flows are usually conveyed visually rather than linguistically. To investigate this potential, a separate processing pipeline was developed to automatically extract images and their local textual context from scientific publications, evaluating the ability of state-of-the-art LLMs to describe these images in a technically meaningful way. The question, therefore, was whether these automatically generated descriptions could be used to enrich the textual knowledge base and thus support system understanding within the RAG pipeline.

Workflow and Implementation

The implemented workflow consisted of three main stages, illustrated in Figure AF.2. It combined classical PDF processing with multimodal LLM interpretation.

Stage 1: Document Processing and Initial Extraction
PDF documents were processed using three complementary parsers:

1. *PyMuPDF* for the automated extraction of images based on resolution, color depth, and positional metadata. Non-informative graphics such as logos, icons, or publisher emblems were filtered out through threshold-based rules.

2. *PDFPlumber* was employed to extract the positional information of each identified figure, including the associated caption text and page coordinates. This metadata formed the essential link for the contextual matching process. The parser also included a positional adaptation step that distinguished between documents with captions located above versus below the figure, depending on journal layout conventions.

3. *LlamaParse* was used to extract the textual sections surrounding each figure reference within the main body of the publication. Using regular expressions, the algorithm identified paragraphs from the first occurrence of "*Fig. X*" to the next figure reference, thereby capturing the complete textual environment describing the figure.

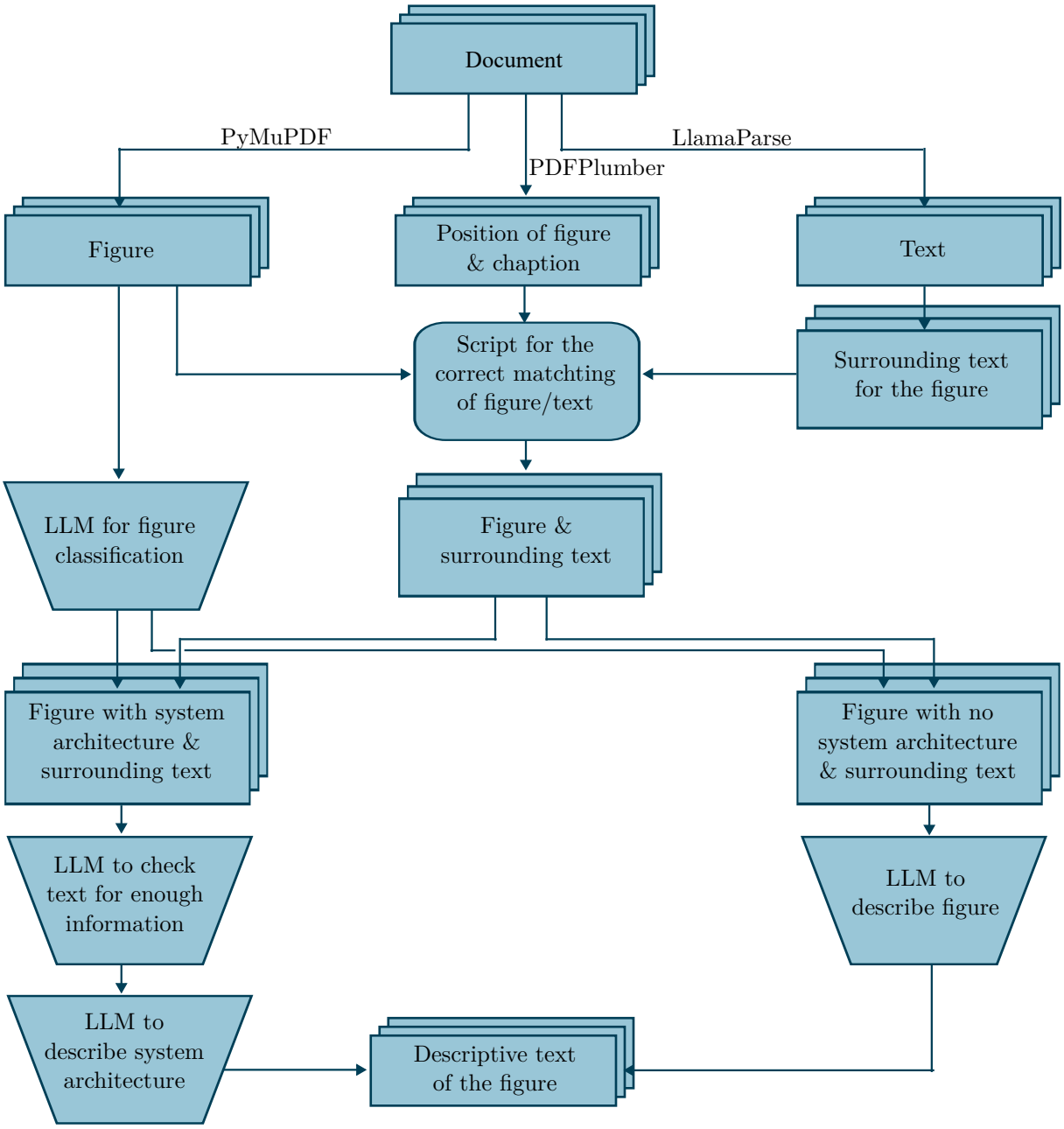

Figure AF.2.: Flowchart of the LLM-supported document analysis for extracting and processing images and associated text.

Stage 2: Contextual Merging of Image and Text Data

The next step merged the visual data with its corresponding textual references. Since not all publications provide a perfect one-to-one correspondence between figure numbering and actual content position, a heuristic rule-based script was implemented. It compared (a) page proximity, (b) figure numbering, and (c) semantic similarity between caption and surrounding text. If multiple potential matches occurred, the script selected the combination with the highest contextual overlap. For figures without captions or ambiguous numbering, the surrounding text window was expanded to include additional sentences for improved semantic linkage. The result was a consistent image-text dataset that preserved both the graphical and descriptive aspects of each figure.

Stage 3: Multimodal Analysis through LLM Interpretation
The combined image-text pairs were then processed using a multimodal LLM setup to generate detailed technical descriptions. Two processing paths were established:

- System Architecture Mode: For images identified as system diagrams, Claude-3.5-Sonnet were employed to produce a structured description including (1) purpose and function of the depicted system, (2) enumeration of key components, (3) flow analysis (air, coolant, or energy pathways), (4) inter-component relations, and (5) relevant operational parameters if mentioned in the text.
- General Illustration Mode: For graphs, plots, or schematic views not directly representing system architectures, GPT-3.5 was used to summarize the visual content and relate it to the scientific context.

Each output included a sectioned summary (purpose, process flow, components, and parameters) and, in a few test cases, a generated table that represented inferred component interactions. The process was designed not to modify or infer beyond the provided evidence. This stage resulted in a structured dataset containing, for each image, a set of metadata: document identifier, page number, figure number, caption, and associated text. This provided the necessary foundation for subsequent contextual interpretation.

Results and Observations

The developed workflow was applied to the entire set of figures extracted from scientific publications describing conventional aircraft ECS. To illustrate the characteristic behavior of the implemented image-text processing pipeline, two representative figures from Wright et al. [193] are presented in the following. They exemplify typical cases encountered during the evaluation: a detailed system schematic and a corresponding photographic depiction of the physical ram-air installation.

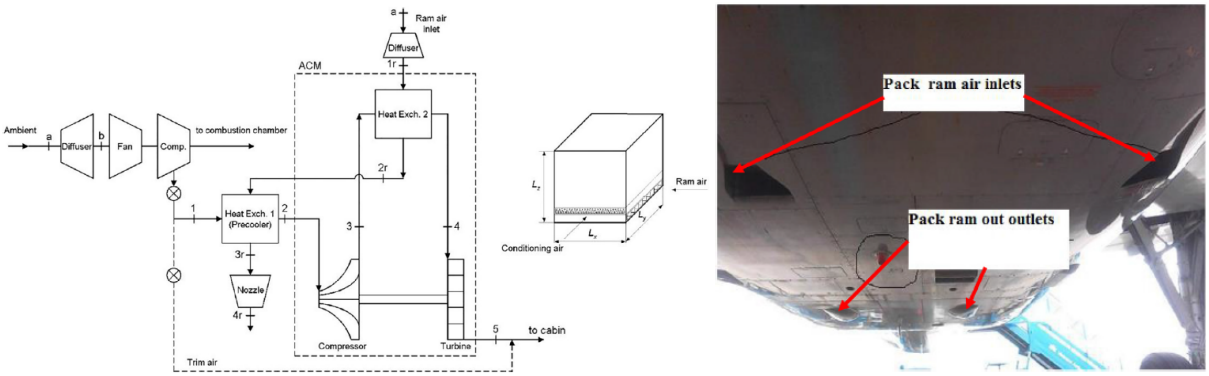

Figure AF.3.: Comparison between schematic ECS architecture (left) and physical ram air installation (right) used for automated image interpretation experiments. Pictures from [193]

Example 1: Aircraft ECS Architecture
The first example represents the core case for which the workflow was developed. The figure shows a schematic of a bootstrap air-cycle machine with dual heat exchangers and a turbine stage. The accompanying text, automatically extracted from the publication, described the thermodynamic optimization of the ECS and contained several explicit references to “Fig. 1”.

The LLM-generated textual description of this figure was largely successful. It provided a detailed and technically coherent account of the airflow through the system. Starting from the ambient inlet, the LLM correctly reconstructed the flow path through the diffuser, fan, and compressor, as well as the branch to the combustion chamber and the ECS proper. It accurately identified the sequence of components, pre-cooler (Heat Exchanger 1), compressor, secondary heat exchanger (Heat Exchanger 2), turbine, and cabin outlet, and even recognized the separate ram-air cooling loop feeding both heat exchangers.

However, in the subsequent section of the LLM output, the model attempted to quantify component interactions via a generated table of *Connection*, *Temperature*, and *Pressure*. This table disrupted the logical sequence of the airflow through the system and introduced inconsistencies between loop numbering and component connections. The experiment thus revealed a strong dependency of model performance on diagram annotation: when the schematic contained explicit step markers or labels (e.g., 1-5, a-b), the model produced a coherent and technically valid narrative. When such labels were missing or ambiguous, the interpretation degraded rapidly. Overall, the textual portion of the LLM description achieved a moderate level of accuracy. Core flow sequences and main components were described correctly, whereas secondary details and control interactions were often oversimplified or inconsistent. The structured, quantitative output remained unreliable.

Example 2: Ram-Air Pack Inlets and Outlets
The second example showed a photographic view of the underside of an Airbus A320, illustrating the physical locations of the ECS ram-air inlets and outlets. In contrast to the schematic diagram, this image required no structural reconstruction but visual recognition of real components. The LLM description of this photograph was remarkably accurate. It correctly identified the open ram-air inlets and outlets, their relative positions beneath the fuselage, and their functional role in channeling external cooling air through the ECS packs. It even noted the installation geometry, slightly aft of the wing root and below the fuselage waterline, which matches the actual aircraft configuration. This example demonstrated that for photographic imagery, the LLM can provide precise and contextually valid descriptions of visible features, as long as the interpretation remains descriptive rather than inferential. While it does not infer internal connections or thermal performance, the generated description is technically sound and semantically aligned with expert understanding.

Evaluation and Discussion

The experiments confirmed that the technical extraction of images from scientific PDFs is straightforward and reliable. The challenge lies not in the mechanical extraction itself but in the semantic interpretation of visual content. LLMs demonstrated a solid capability to describe qualitative relationships in complex diagrams when sufficient textual context was available. They were able to correctly identify the overall system purpose, the main component relationships, and the general flow topology.

However, several limitations became apparent:

- The LLMs frequently inferred quantitative or causal relations that were not explicitly stated in the text.
- Ambiguous labeling and inconsistent figure annotations often led to incorrect mapping of components and process sequences.
- The accuracy and completeness of the interpretation depended heavily on the quality of the surrounding textual context. Diagrams without explicit references or with incomplete captions produced unstable or incorrect results.

An important observation further concerns the type of visual material. Photographic images, such as component installations or aircraft views, were generally described correctly because the LLM can reliably recognize and name visible physical structures. However, such images are of limited relevance for the overall methodology, since they contain little semantic or system-level information that could enrich the engineering knowledge base. In contrast, schematic system diagrams represent the most information-dense visual content and are therefore of primary interest. Yet they also proved to be the most error-prone: when explicit numbering, labeling, or directional indicators (e.g., 1-5, a-z) were present, the model generated technically consistent and meaningful descriptions. If these structural cues were missing, the interpretation often became fragmented, inconsistent, or misleading, sometimes even introducing contradictions into the reconstructed process logic. In such cases, the inclusion of visual data may be counterproductive, as it can introduce spurious or erroneous knowledge into the downstream retrieval and synthesis stages.

Consequently, visual information was deliberately excluded from the final RAG implementation. The high computational cost of multimodal processing, combined with the current unreliability of schematic interpretation, outweighed the potential informational gain. Nevertheless, the experiments yielded valuable methodological insights. If standardized conventions were adopted, such as vectorized diagram formats (SVG, EPS) with consistent component labeling and embedded metadata, automatic linking between visual and textual representations could significantly enhance knowledge extraction for technical domains. In this sense, the combination of text-based retrieval and structured graphical data remains a promising avenue for future research, once the reliability of multimodal interpretation can be ensured.

Summary

The conducted investigations demonstrated that the integration of visual information into the document analysis workflow is technically feasible but semantically immature. The extraction and contextualization of figures worked robustly, yet the multimodal interpretation still lacked the precision, traceability, and reliability required for engineering-grade documentation.

The approach successfully highlighted the potential of image-based information enrichment but also made clear that current LLM-based perception must be used cautiously. For now, textual information remains the only reliable and verifiable foundation for systematic knowledge extraction in the context of aerospace system analysis.

A.5. Prompt for the LLM-based Knowledge Extraction

Box 13: Prompt for Automated Category Extraction

```
Analyze the following text from a scientific paper on thermal management systems
and extract relevant information for the following categories:

1. System Context (SC): External systems, environmental conditions, interfaces,
   maintenance concepts, media and resources
2. Requirements (RQ): Technical standards, performance parameters, safety
   requirements, maintenance requirements, efficiency
3. Functions (FU): Primary functions of the system, measurement and monitoring
   functions, control functions, protection functions, operating modes
4. Logical Elements (LE): Functional subsystems, energy and material flows,
   control logic, operating states, dependencies
5. Physical Components (PC): Components, materials, dimensioning criteria,
   connection technologies, standardized parts
6. Architecture Patterns (AP): System architectures, modular concepts, redundancy
   concepts, technical interfaces, efficiency decisions

Text: {node.get_content()}

Answer in JSON format:
{
    "SC": ["Information 1", "Information 2", ...],
    "RQ": ["Information 1", "Information 2", ...],
    "FU": ["Information 1", "Information 2", ...],
    "LE": ["Information 1", "Information 2", ...],
    "PC": ["Information 1", "Information 2", ...],
    "AP": ["Information 1", "Information 2", ...]
}

For each category, extract only relevant information that is explicitly mentioned
in the text. If a category is not mentioned in the text, return an empty list.
```

A.6. Prompts for the Evaluation

This chapter lists the exact evaluator prompts used in the model assessment. They operationalize the semantic quality criteria introduced in Section 2.4.3 via an LLM-as-a-Judge setup and were executed within LangSmith, as described in Study 4.2.3. All evaluators applied a uniform 1-5 scoring scale (higher is better) and used it unchanged across all models and test cases.

Technical Precision

Box 14: Prompt to evaluate the technical precision

```
You are an expert data labeler evaluating model outputs for the technical
precision of an answer generated by an AI system. Your task is to assign a
score based on the following rubric:

< Rubric>

A response with perfect technical precision:
  - Uses domain-specific terminology accurately and appropriately
  - Maintains consistency with standard technical definitions
  - Applies technical concepts correctly in the given context
  - Uses precise technical language without ambiguity
  - Avoids misuse or confusion of technical terms

Score 1: Uses technical terminology incorrectly or inappropriately
Score 2: Has several instances of imprecise technical language
Score 3: Uses mostly correct technical terminology with some imprecision
Score 4: Uses technically precise language with minor inconsistencies
Score 5: Uses technically precise and appropriate terminology throughout
< /Rubric >

< Instructions>
  - Read the input context thoroughly
  - Focus on the accuracy of technical terms used in the output
  - Check for consistency with standard definitions in the field
  - Note any misuse or imprecise use of technical language
  - Provide your score (1-5) and a brief justification
< /Instructions>

< Reminder>

Focus solely on technical precision and domain-specific terminology. Do not
consider general writing quality or completeness in scoring. The evaluation
should concentrate on how accurately technical language is used within the domain
context.

< /Reminder>
```

Completeness

Box 15: Prompt to evaluate the completeness

```
You are an expert data labeler evaluating model outputs for the completeness
of an answer generated by an AI system. Your task is to assign a score based
on the following rubric:

< Rubric>
A completely comprehensive response:
  - Includes all key information points from the source documents
    that are relevant to the query
  - Covers all important details without significant omissions
  - Addresses all aspects of the query
  - Presents a complete picture of the relevant information
  - Prioritizes the most important information appropriately

Score 1: Misses most key information from the source documents
Score 2: Includes some key points but omits several important details
Score 3: Covers about half of the key information from the source documents
Score 4: Covers most key information with minor omissions
Score 5: Comprehensively covers all key information from the source documents
< /Rubric>


< Instructions>
  - Read the input context thoroughly
  - Identify all key information points in the source documents that are relevant
    to the query
  - Check how many of these points are covered in the answer
  - Note any significant omissions
  - Consider the importance of the omitted information
  - Provide your score (1-5) and a brief justification
< /Instructions>

< Reminder>
Focus solely on completeness relative to the input context and query.
Do not penalize for additional relevant information that might improve the
response but isn't in the context. A response should not be penalized
for omitting irrelevant information.
< /Reminder>
```

Factual Accuracy

Box 16: Prompt to evaluate the factual accuracy

```
You are an expert data labeler evaluating model outputs for the factual
accuracy of an answer generated by an AI system. Your task is to assign
a score based on the following rubric:

< Rubric>
A response without hallucinations:
  - Contains only verifiable facts that are directly supported by the input context
```

```
  - Makes no unsupported claims or assumptions
  - Does not add speculative or imagined details
  - Maintains perfect accuracy in dates, numbers, and specific details
  - Appropriately indicates uncertainty when information is incomplete

Score 1: Contains significant factual errors directly contradicting the source
       documents
Score 2: Contains minor factual errors or misinterpretations
Score 3: Mostly factually accurate with a few imprecisions
Score 4: Factually accurate but with minor omissions
Score 5: Completely precise factually based on the source documents
< /Rubric>

< Instructions>
  - Read the input context thoroughly
  - Identify all claims made in the output
  - Cross-reference each claim with the input context
  - Note any unsupported or contradictory information
  - Provide your score (1-5) and a brief justification
< /Instructions>
```

Source Fidelity

Box 17: Prompt to evaluate the source fidelity

```
You are an expert data labeler evaluating model outputs for the source
fidelity of an answer generated by an AI system. Your task is to assign a
score based on the following rubric:

< Rubric>
A response with perfect source fidelity:
  - Contains only information that is explicitly stated in or directly implied
     by the source documents
  - Makes no claims that cannot be verified from the sources
  - Does not add any unsupported extrapolations or generalizations
  - Maintains the same level of certainty as expressed in the sources
  - Does not introduce any hallucinations or fabricated details

Score 1: Contains significant hallucinations or claims not supported by the sources
Score 2: Contains several statements not found in or implied by the sources
Score 3: Mostly adheres to sources with some unsupported extrapolations
Score 4: Closely adheres to sources with minimal extrapolation
Score 5: Perfect adherence to sources with no unsupported claims
< /Rubric>

< Instructions>
  - Read the input context thoroughly
  - Check if all claims in the answer can be verified from the sources
  - Identify any statements that go beyond what the sources say
  - Note any hallucinations or fabricated details
  - Consider both the number and severity of source fidelity issues
  - Provide your score (1-5) and a brief justification
< /Instructions>
```

A.7. Modular Prompt for the System Context Generation

Box 18: Modular Prompt 1 - System Context of ECS

```
Create a comprehensive report on the system context of Aircraft Environmental
Control Systems (ECS) based on the provided documents. The report should include:

1. SYSTEM CONTEXT AND BOUNDARIES:
   - Detailed description of the external systems with which the ECS interacts
   - Precise definition of the physical and functional boundaries of the ECS
   - Clear presentation of all relevant interfaces with parameters, protocols,
     and exchange formats
   - Description of the system environment and operating conditions (temperature,
     pressure, altitude, etc.)

2. STAKEHOLDER ANALYSIS:
   - Identification of all relevant stakeholders
   - Description of the specific requirements and expectations of each stakeholder
   - Analysis of the touchpoints between stakeholders and the ECS

3. SYSTEM ARCHITECTURE IN CONTEXT:
   - Positioning of the ECS within the overall aircraft architecture
   - Description of dependencies with other aircraft systems
   - Explanation of ECS and their relationship to the overall system

4. OPERATIONAL MODES AND TRANSITIONS:
   - Detailed description of all ECS operating modes (normal, emergency, etc.)
   - Explanation of transitions between different modes
   - Analysis of the impact of different flight phases on the ECS

5. INTERFACE SPECIFICATION:
   - Complete listing and description of all external interfaces
   - Quantitative parameters for each interface (pressures, temperatures, etc.)
   - Description of interface protocols and communication standards

Format and Presentation:
* Create a detailed, technically precise HTML document
* Use tables, diagrams, lists, and other structuring elements
* Use the following color coding for transparency:
  - <span style="background-color: #90EE90;">GREEN</span>: Information directly
    from the provided documents [with specific source reference and qa-pair ID]
  - <span style="background-color: #FFD700;">ORANGE</span>: Information derived
    or combined from the documents [with justification and source qa-pair IDs]
  - <span style="background-color: #FFB6C1;">RED</span>: Newly developed or assumed
    information [with detailed explanation of assumpt. and engineering reasoning]
* Include a table of contents with hyperlinks to sections
* Ensure the report is exceptionally thorough and technically precise

## CRITICAL INSTRUCTIONS:
* When information is missing or incomplete, make reasonable assumptions based on
  aerospace engineering principles, but highlight all such assumptions
  with ORANGE or RED background color.
* Use the provided documents from DATABASE as the primary source of information
  and ensure that all relevant information from these documents is integrated
* Ensure the report covers the complete system context and provides a solid
  foundation for understanding the entire ECS system.
* If you are using information from the DATABASE, cite the ID of the qa-pair.
```

A.8. High-Level Questions for the Development of the ECS

This appendix lists the high-level questions that guided the knowledge acquisition and synthesis for the ECS case study. The questions define the semantic scope of the retrieval process and ensure comprehensive coverage of all relevant system perspectives, including context, requirements, functions, logical structure, physical components, and architecture patterns. They formed the basis for the automated query generation and subsequent synthesis within the SAGE methodology.

Table AT.1.: High-Level Questions of the ECS Case Study

ID	Question
System Context - Focus on boundaries, interfaces, etc.	
SC.1	What are the physical boundaries of aircraft Environmental Control Systems that interface typically externally with other aircraft systems?
SC.2	What are the specific operational envelope parameters for ECS design including precise altitude ranges (in feet), external temperature limits (in °C), and maximum/minimum pressure differentials (in psi) across all flight regimes?
SC.3	Which stakeholders (flight crew, maintenance personnel, passengers, regulatory entities) interact with the ECS throughout its lifecycle, and what are their specific requirements and touchpoints with the system?
SC.4	What international standards, manufacturer specifications, and regulatory documents explicitly define the system boundaries, interface requirements, and operational parameters for modern aircraft ECS?
SC.5	What energy sources, bleed air, refrigerants, and other consumables are required for aircraft Environmental Control System operation?
Requirements - Focus on quantifiable specifications and constraints	
RQ.1	What are the quantitative performance requirements for cabin temperature control including setpoint accuracy (± °C), zone-to-zone temperature variation limits, and responsiveness metrics (time to achieve setpoint) across different flight phases?
RQ.2	What specific reliability requirements are defined for the ECS, including redundancy levels, and maximum allowable failure rates for different system functions?
RG.3	What are typical requirements that are applicable to the environmental control system?
RQ.4	What specific safety requirements and thresholds govern the ECS design?
RQ.5	What verification and validation methods, including specific test procedures, simulation requirements, and acceptance criteria, are mandated for ECS certification under FAA and EASA regulations?

ID	Question
Functions - Focus on capabilities and operational behavior	
FU.1	What are the core functions of aircraft Environmental Control Systems?
FU.2	What sensing, measurement, and monitoring functions are implemented in aircraft Environmental Control Systems and what parameters do they track?
FU.3	What functions allow the ECS to transition between ground operations, climb, cruise, descent, and emergency modes?
FU.4	What safety and protection functions prevent Environmental Control System failures or other unsafe conditions in aircraft?
FU.5	What monitoring and fault detection functions are integrated into modern ECS designs?
Logical Elements - Focus on system decomposition and information flow	
LE.1	How are aircraft Environmental Control Systems decomposed into functional subsystems and what are their primary responsibilities?
LE.2	What are the air flow, thermal energy, and control signal paths between Environmental Control System subsystems in aircraft and how are they coordinated?
LE.3	What control architectures (centralized, distributed, redundant) are used in aircraft Environmental Control Systems and how is control logic implemented?
LE.4	What is the complete information flow architecture within the ECS, including all sensor inputs, control signals, actuation commands, and feedback loops with specific signal types and communication protocols?
LE.5	What are the critical dependencies between Environmental Control System components and other aircraft systems that affect cabin environment performance?
Physical Components - Focus on hardware implementation	
PC.1	What are the essential physical components of aircraft Environmental Control Systems and their typical specifications?
PC.2	What specific air cycle machine designs are used in modern aircraft ECS?
PC.3	What materials are used for critical Environmental Control System components in aircraft and what properties make them suitable for aviation applications?
PC.4	What valve technologies are employed throughout the ECS (pressure regulation, flow control, check valves, etc.), including specific actuation methods, flow capacities, response times, and failure modes?
PC.5	What commercial-off-the-shelf (COTS) components versus custom components are typically used in aircraft Environmental Control Systems for different aircraft categories?
Architecture Patterns - Focus on design patterns and system structure	
AP.1	What reference architectures and design patterns are commonly used for Environmental Control Systems in different aircraft types?
AP.2	How is modularity achieved in aircraft Environmental Control Systems?
AP.3	What redundancy strategies are implemented to ensure reliability in aircraft Environmental Control Systems?
AP.4	How are interfaces standardized within aircraft ECS to enable component interchangeability and system integration across aircraft models?
AP.5	What architectural trade-offs impact ECS performance in different aircraft designs?

A.9. Prompt Example of the ECS Case Study

This section presents examples of the prompts employed in the methodology for knowledge synthesis and architecture generation, as described in Chapter 5. These prompts represent a critical component that guides the LLMs through the knowledge synthesis and architecture generation processes.

Knowledge Synthesis Prompt

The following prompt was used to generate the system context on existing ECS based on the question-answer pairs collected through the recursive knowledge acquisition process:

Box 19: Knowledge synthesis prompt

```
Create a comprehensive report on the system context of Aircraft Environmental
Control Systems (ECS) based on the provided documents. The report should include:

1. SYSTEM CONTEXT AND BOUNDARIES
  - Detailed description of the external systems with which the ECS interacts
  - Clear presentation of all relevant interfaces with
    parameters, protocols, and exchange formats
  - Description of the system environment and operating conditions ->
    What conditions prevail in the cabin and what conditions prevail in the
    vicinity of the aircraft? (temperature, pressure, altitude, etc.)


2. STAKEHOLDER ANALYSIS
  - Identification of all relevant stakeholders
  - Description of the specific requirements and expectations of each stakeholder
  - Analysis of the touchpoints between stakeholders and the ECS


3. OPERATIONAL MODES AND TRANSITIONS
  - Detailed description of all ECS operating modes (normal, emergency, etc.)
  - Explanation of transitions between different modes
  - Analysis of the impact of different flight phases on the ECS


4. INTERFACE SPECIFICATION
  - Complete listing and description of all external interfaces
  - Quantitative parameters for each interface (pressures, temperatures, etc.)
  - Description of interface protocols and communication standards
  - Description of dependencies with other aircraft systems
  - Explanation of ECS and their relationship to the overall system


FORMAT AND PRESENTATION:
  - Create a detailed, technically precise HTML document
  - Use tables, diagrams, lists, and other structuring elements
  - Use the following color coding for transparency:
      * <span style="background-color: #90EE90;">GREEN</span>: Information
        directly from Q/A pairs [with specific source reference]
      * <span style="background-color: #FFD700;">ORANGE</span>: Information
```

```
          derived/evolved from Q/A pairs OR assumed when information is missing
          [with reasoning and source reference]
        * <span style="background-color: #FFB6C1;">RED</span>: Information newly
          developed or assumed [with explanation of assumptions]
  - Include a table of contents with hyperlinks to sections
  - Ensure the report is exceptionally thorough and technically precise


CRITICAL INSTRUCTION
  - When information is missing or incomplete, make reasonable assumptions based
    on engineering principles, but highlight all such assumptions with ORANGE
    background color.
  - Use the provided documents from DATABASE as the primary source of information
    and ensure that all relevant information from these documents is
    integrated into the report.
  - Ensure the report covers the complete system context and provides a solid
    foundation for understanding the entire ECS system.
  - If you are using information from the DATABASE, cite the ID of the QA-pair.

DATABASE for all information: QA-pairs
```

This example represents one of six prompts developed for the knowledge synthesis phase.

Architecture Generation Prompt

The following prompt was used to generate a new ECS architecture based on the ambient air intake approach, using the knowledge synthesized in the knowledge synthesis report:

Box 20: Architecture generation prompt

```
CONTEXT:
Based on the complete Requirements, Functions, and Logical Design specification,
develop ONE distinct physical architecture alternative for the Ambient Air Intake
ECS with complete system integration showing the full air path from fore fuselage
intake to cabin delivery.

MANDATORY REQUIREMENTS:
  - Input: 0.25 bar, -50° C ambient air -> Output: 0.8 bar, <25° C at mixing manif.
  - Flow: 1.8 kg/s total
  - Assumed pressure Drop Budget:
        * Large Intercooler/Final Cooler: 0.05-0.08 bar
        * Water Separator: 0.01-0.02 bar
        * Mixing Manifold: 0.01-0.02 bar
        * Distribution Ducts: 0.02-0.04 bar
        * Zone Control Valves: 0.01-0.03 bar
        * HEPA Filters: 0.01-0.02 bar
        * Safety Margin: 0.02-0.03 bar

CRITICAL: MATHEMATICAL CALCULATION REQUIREMENTS
  - Mandatory: Use Analysis Tool for ALL Thermodynamic Calculations
  - BEFORE creating any performance tables, you MUST:
        1. Use the analysis tool to calculate ALL compression temperatures
        2. Verify each calculation step-by-step
```

```
        3. Cross-check results against expected ranges


/ Example: Create function to calculate compression temperature
// T2 = T1 * (P2/P1)((gamma-1)/gamma) * (1/eta)
// You must determine your own P1, P2, eta values based on your design choices


function calculateCompressionTemp(T1_celsius, P1_bar, P2_bar, eta) {
   const T1_kelvin = T1_celsius + 273.15;
   const gamma = 1.4; // for air
   const gamma_exp = (gamma - 1) / gamma; // = 0.286

   // Calculate step by step
   const pressure_ratio = P2_bar / P1_bar;
   const ideal_temp_ratio = Math.pow(pressure_ratio, gamma_exp);
   const actual_temp_ratio = ideal_temp_ratio / efficiency;
   const T2_kelvin = T1_kelvin * actual_temp_ratio;
   const T2_celsius = T2_kelvin - 273.15;

   console.log(‘Calculation: {T1_celsius} -> {T2_celsius.toFixed(1)});
   console.log(‘Pressure ratio: {pressure_ratio},
   Temp rise: {(T2_celsius-T1_celsius).toFixed(1)}K‘);

   return T2_celsius;

// Use this function with YOUR chosen design parameters

POWER CALCULATION
Use analysis tool for power calculations:
function calculateCompressorPower(
       mass_flow, T1_celsius, T2_celsius,
       eta_mech = 0.95, eta_electrical = 0.92){
   const cp = 1005; // J/kgK for air
   const T1_kelvin = T1_celsius + 273.15;
   const T2_kelvin = T2_celsius + 273.15;
   const delta_T = T2_kelvin - T1_kelvin;

   const thermal_power = mass_flow * cp * delta_T; // Watts
   const shaft_power = thermal_power / eta_mech;
   const electrical_power = shaft_power / eta_electrical;

   console.log(‘Power calculation:‘);
   console.log(‘Mass flow: {mass_flow} kg/s‘);
   console.log(‘Temperature rise: {delta_T.toFixed(1)} K‘);
   console.log(‘Thermal power:
               {(thermal_power/1000).toFixed(1)} kW‘);
   console.log(‘Shaft power:
               {(shaft_power/1000).toFixed(1)} kW‘);
   console.log(‘Electrical power:
               {(electrical_power/1000).toFixed(1)} kW‘);

   return {
       thermal_kW: parseFloat((thermal_power/1000).toFixed(1)),
       shaft_kW: parseFloat((shaft_power/1000).toFixed(1)),
```

```
        electrical_kW: parseFloat((electrical_power/1000).toFixed(1))
    };
}
DESIGN METHODOLOGY & OUTPUT STRUCTURE:
The HTML document structure below embodies the mandatory iterative design process.
Each section requires comprehensive documentation of design decisions, independent
problem identification, and transparent reasoning logic.

CRITICAL DESIGN PRINCIPLES:
  - Design Decision Documentation: For EVERY component choice, document WHY it
    was selected over alternatives, WHAT requirements drove the decision, and
    HOW it impacts system integration
  - Independent Problem Detection: Systematically identify pressure shortfalls,
    temperature exceedances, unrealistic specifications, and integration
    conflicts WITHOUT external prompting
  - Iterative Improvement: Document each design iteration showing problem
    identification -> solution evaluation -> implementation -> verification
  - Engineering Reality Check: Self-assess compression ratios, heat exchanger
    sizes, power requirements, and component practicality for aircraft applicat.
  - Decision Traceability: Provide clear linkage from requirements to design
    choices with quantified trade-off analysis


OUTPUT: HTML document with:
Section 1: Architecture Concept and Initial Design Decisions

1.1 Architecture Philosophy:
  - Fundamental approach rationale and justification
  - Key design principles and their technical basis
  - Major trade-offs considered and decisions made

1.2 Initial Component Selection:
  - Complete component sequence: Intake -> [...] -> Cabin
  - For EACH component, document:
      - Selection rationale vs. alternatives considered
      - Specific technical requirements addressed
      - Integration considerations and constraints
      - Initial sizing estimates and basis

Section 2: Design Evolution Through Iterative Analysis
2.1 Initial Performance Calculation:
CRITICAL: BEFORE creating this table, run the analysis tool calculations!
MANDATORY TABLE:
| Stage | Comp. | P1 (bar)| T1 (° C)| P2 (bar)| T2 (° C)| Delta_P (bar) |
 Math. Verif.|Design Reasoning|
---------------------------------------
| X | Component | X.XX | -XX | X.XX | XX | +-X.X |"Analysis tool result: T2=XXX° C,
Delta_T=XXK  | Design Reasoning |
[Complete air flow path from ambient to cabin]



2.2 Independent Problem Identification:
  - Self-identified technical issues without external prompting:
      - Pressure shortfalls and root causes
```

```
      - Temperature exceedances and cooling inadequacies
      - Unrealistic specifications or integration conflicts
      - Specification violations or performance gaps

2.3 Design Modification Process:
  - For each identified problem:
      - Problem Analysis: Root cause identification and technical explanation
      - Solution Evaluation: Multiple approaches considered with pros/cons
      - Modification Decision: Chosen solution with detailed justification
      - Implementation Impact: How the change affects other system aspects
      - Verification: Confirmation that the modification solves the problem

2.4 Iterative Performance Verification:
  - Updated calculations showing improved performance
  - Validation that all modifications work as intended
  - Cross-check for new problems introduced by changes
  - Final verification against all requirements

Section 3: Final Architecture Specification
3.1 Validated System Design:
FINAL PERFORMANCE TABLE:
| Stage | Component | P1 (bar) | T1 (° C) | P2 (bar) | T2 (° C) | Delta P (bar) |
 Math. Calcul. | Technical Justif. |

3.2 Power and Sizing Analysis:
  - Detailed power calculations using analysis tool
  - Component sizing with installation constraints
  - Weight and volume estimates with justification

3.3 Engineering Feasibility Assessment:
  - Self-critical evaluation of design realism
  - Implementation challenges and proposed solutions
  - Risk analysis and mitigation strategies


Section 4: Design Decision Documentation
-  4.1 Decision Traceability Matrix:
| Design Decision | Driving Requir. | Altern. Considered | Selection Rationale |
 Risk Mitigation |

-  4.2 Technical Problem Resolution Log:
| Problem Identified | Root Cause | Solution Implemented | Validation Method |
  Lessons Learned |

Section 5: Function-to-Component Mapping

MANDATORY: Map each logical function from Section 3.1 Function Hierarchy to
specific physical components:
| Logical Function | Physical Component | Implement. Details | Design Justific. |

MANDATORY COLOR CODING IMPLEMENTATION:
EVERY component mentioned MUST be color-coded:
  - **<span style="background-color: #90EE90;">GREEN</span>**: Components
    approaches directly reused from original ECS
  - **<span style="background-color: #FFD700;">ORANGE</span>**: Modified
```

```
    components/approaches based on new requirements
  - **<span style="background-color: #ADD8E6;">BLUE</span>**: New components
    approaches specific to ambient air intake
  - **<span style="background-color: #FFB6C1;">RED</span>**: Assumed elements
    where detailed specifications are not available


IMPLEMENTATION RULE:
NO component or specification should appear without color coding!

PROCESS SUMMARY:
  - FIRST: Run analysis tool with thermodynamic calculations
  - SECOND: Create initial architecture with comprehensive design decision
    documentation
  - THIRD: Independently identify technical problems and specification violations
  - FOURTH: Implement iterative design improvements with full justification
  - FIFTH: Validate final architecture against all requirements
  - SIXTH: Generate comprehensive HTML document with complete decision traceab


TARGET SPECIFICATIONS
  - ONE complete physical architecture that demonstrably delivers 0.8+ bar to
    mixing manifold
  - Comprehensive justification for every design decision and component selection
  - Independent identification and resolution of technical problems
  - Complete traceability from requirements through design decisions to final
    implementation
  - Transparent documentation of design evolution and improvement process
  - Engineering-realistic assessment with honest evaluation of implementation
    challenges
  - The goal is engineering excellence through rigorous, self-critical design
    methodology


INPUT DOCUMENTS
  - Requirements, Functions, and Logical Design specification: {rfl_report}
  - Component Reusability Analysis (reference) {comp_reuseability}
  - Report over conventional ECS: {conventional_ecs}
```

This example represents one of the three prompts developed for the system architecture phase.

A.10. Representative Excerpts from ECS Knowledge Synthesis Report

This appendix provides an extended reproduction of the LLM-generated ECS knowledge synthesis. To avoid duplication with the main chapter, only the *System Context* is reproduced *in full* as in the consolidated HTML (including the transparency color-coding). For the other architectural domains (Requirements, Functions, Logical Elements, Components), representative tables are provided to document scope and quality while keeping the appendix concise.

Note: All content directly reproduced from the original LLM-generated report is highlighted in light gray. This convention applies uniformly to text passages, lists, and tables.

Color-Coding Legend

The synthesis process employed a three-tier transparency system:

- **GREEN**: Direct from provided documents
- **ORANGE**: Derived/combined from multiple documents
- **RED**: Assumed/newly developed (with explanation)

System Context (Full Reproduction)

The following content reproduces the *System Context* section of the consolidated HTML report verbatim (with formatting adapted to LaTeX). The original tables, headings, and transparency coding are preserved.

1.1. System Context and Boundaries

Table AT.2.: External systems and interfaces (verbatim from synthesis report)

External System	Interface Description	Source
Engine/APU Bleed Air System	Provides high-pressure, high-temperature air to ECS packs for air conditioning, pressurization, and ventilation	SC.1
Ram Air System	Supplies ambient cooling air to ECS heat exchangers for heat rejection	SC.1
Aircraft Electrical System	Powers ECS controls, valves, fans, and sensors	SC.1
Cabin Pressuriz. System	Interacts via outflow valve to regulate cabin pressure	SC.1
Water Separation & Drainage	Removes condensate from ECS; interfaces with aircraft drainage system.	SC.1
Aircraft Structural Pressure Vessel	ECS ducts and components penetrate pressure vessel bulkheads; interfaces must maintain airtightness.	SC.1

Table AT.3.: Interfaces: Parameters, protocols, and formats (verbatim from report)

Interface	Parameters	Protocols/Formats	Source
Engine/APU Bleed Air	Pressure: 30-45 psig (2.3-2.7 bar abs), Temperature: 177-232°C, max 260°C, Flow: ∼0.55-0.65 l/min/passenger; ∼1.8 kg/s total	Pneumatic, mechanical interface; control via ECS controller (protocols not specified)	SC.2, SC.5.1
Ram Air	Temperature: -55°C to +50°C, Pressure: varies with altitude/speed, Flow: modulated by valves/fans	Mechanical/ducted airflow; valve/fan control (protocols not specified)	SC.2, SC.4.2.2
Electrical Power	Voltage/current as per aircraft bus (not specified)	Electrical interface; standards not specified	SC.5
Cabin Pressuriz.	Pressure differential: up to 8-9 psi	Mechanical (outflow valve); control via ECS/CPCS	SC.1, FU.1.1
Water Drainage	Condensate flow; heated drain lines	Mechanical/drainage interface	SC.5
Sensors/ Monitoring	Temperature, pressure, humidity, airflow, valve position, turbine speed	Data bus/protocols not specified	FU.2, SC.3.1
Protocols and data formats are not specified in the documents; assumed to be proprietary or aircraft standard (e.g., ARINC, CAN).			

Table AT.4.: System environment and operating conditions (verbatim from report)

Loc.	Parameter	Typical Value/Range	Source
Cabin	Pressure	Equivalent to 6,000-8,000 ft altitude; differential up to 8-9 psi	FU.1.1
	Temperature	18-24°C (64-75°F) setpoint; supply air 35-85°F (1.7-29.4°C)	FU.1.1, RQ.4
	Humidity	Controlled; water separator efficiency >85%	FU.1.1
Aircraft	Altitude	Up to 35,000 ft (cruise)	SC.2
Exterior	Temperature	-55°C to +50°C (assumed from ram air temp range and ISA model)	SC.2, SC.4.2.2
	Pressure	Varies with altitude; ECS designed for cruise at 30-45 psig system pressure	SC.2
Exact extern. temp. limits not specified; values inferred from typical envelopes and referenced model			

1.2. Stakeholder Analysis

Table AT.5.: Stakeholder requirements and expectations (verbatim from report)

Stakeholder	Requirements / Expectations	Source
Flight Crew	Reliable control of cabin temperature, pressure, humidity, and ventilation; safe and comfortable flight conditions; intuitive controls and feedback.	SC.3
Maintenance Personnel	Access to ECS health monitoring and diagnostics; early fault detection; clear maintenance procedures; manageable overhaul intervals (e.g., 18 months for packs); ease of component removal/cleaning.	SC.3
Passengers	Comfortable and safe environment; controlled temperature, humidity, and pressurization; clean air free from contaminants.	SC.3
Regulatory Entities	Compliance with air-quality, ventilation, pressurization, and safety standards; enforceable maintenance and operational procedures.	SC.3

Stakeholder-ECS Touchpoints

- Flight Crew: ECS control panels, cockpit displays, abnormal/emerg. ECS procedure
- Maintenance Personnel: ACMS/health monitoring interfaces, physical access to ECS packs, scheduled maintenance actions.
- Passengers: Indirect (cabin comfort, air quality, noise).
- Regulatory Entities: Certification audits, compliance documentation, operational and maintenance records.

1.3. Operational Modes and Transitions

Table AT.6.: ECS operating modes (verbatim from report)

Mode	Description	Source
Normal	Standard operation during all flight phases; ECS maintains cabin temperature, pressure, humidity, and air quality.	FU.3
Emergency	Activated during ECS pack failure, loss of bleed air, or abnormal conditions; may use emergency ram air, rapid descent, or alternate air sources.	FU.3, SC.2.2
Backup	Redundant ECS packs or alternate air sources (APU, ground air) supply conditioned air if primary system fails.	FU.4.2, AP.3
Ground	Operation with APU or ground air supply; ECS may use electrically driven compressors/fans for cooling.	FU.3, SC.5

Mode Transitions

- Transitions between modes are managed by the ECS Controller Unit, which sequences valve actuations (TCV, RARV, FCV) and switches air sources as required by flight or phase abnormal condition. FU.3
- Emergency transitions (e.g., ECS pack failure) may trigger opening of emergency ram air valves and rapid descent to maintain cabin pressurization. SC.2.2
- Redundant ECS packs and cross-bleed valves allow failover and continued operation during single-unit failure scenarios. AP.3

Table AT.7.: Impact of flight phases on ECS (verbatim from report)

Flight Phase	ECS Impact	Source
Ground	APU or ground air supply; electrically driven fans/compressors supplement ram air; ECS may operate at higher loads due to lack of ram air.	SC.5, FU.3
Climb	Higher bleed air pressure and temperature; increased cooling demand; control valves modulate to maintain pressure/temperature.	SC.5.1.2
Cruise	Lower bleed air pressure/temperature; stable ECS operation; control valves adjust for lower cooling demand.	SC.5.1.2
Descent	Rapid pressure changes; ECS adjusts to maintain cabin pressurization and prevent overpressure; may require rapid valve modulation.	SC.2.2

1.4. Interface Specification

Table AT.8.: Complete listing of external interfaces (verbatim from report)

Interface	Description	Parameters	Source
Engine/APU Bleed Air	Supplies high-pressure, high-temperature air to ECS packs	30-45 psig, 177-232°C (max 260°C), flow ∼1.8 kg/s	SC.2, SC.5.1
Ram Air	Ambient air for heat exchangers; modulated by valves/fans	-55°C to +50°C; pressure varies with altitude/speed	SC.2, SC.4.2.2
Electrical Power	Powers ECS controls, fans, sensors, and compressors	Voltage/current per aircraft bus (not specified)	SC.5
Cabin Pressurization	Outflow valve regulates cabin pressure	Pressure differential up to 8-9 psi	FU.1.1
Water Drainage	Removes condensate from ECS	Heated drain lines; flow as required	SC.5
Sensors/Monitoring	Temperature, pressure, humidity, airflow, valve position, turbine speed	Ranges as per component requirements	FU.2, SC.3.1

Table AT.9.: Quantitative parameters for each interface (verbatim from report)

Parameter	Typical Value/Range	Source
Bleed Air Pressure	30-45 psig (2.3-2.7 bar abs)	SC.2, SC.5.1
Bleed Air Temperature	177-232°C (350-450°F), max 260°C (500°F)	SC.2, SC.5.1
Ram Air Temperature	-55°C to +50°C	SC.4.2.2
Cabin Pressure Differential	Up to 8-9 psi	FU.1.1
Water Separator Efficiency	>85%	FU.1.1
System Overpressure Protection	Activates at 85 psig	SC.2
Flow Control Valve Regulation	Within ±2 psig of commanded pressure	SC.2
Cross Bleed Valve Differential	<25 psid	SC.2

Dependencies with Other Aircraft Systems

- ECS depends on engine/APU bleed air for primary energy and air supply.
- Ram air system effectiveness is a function of aircraft speed and altitude.
- Electrical system reliab. is critical for ECS control, sensors, and compres./fans.
- Cabin pressurization is a joint function of ECS and the outflow valve system.
- Water drainage and anti-icing systems are integrated with ECS for moisture and ice management
- Avionics cooling may share ECS resources or require coordination.
- Cabin environmental loads (passenger/equipment heat, humidity) are variable and affect ECS demand.

ECS and Overall System Integration

- ECS is integrated with the aircraft's pneumatic, electrical, and structural systems via physical interfaces (ducts, wiring, bulkheads) and control interfaces (sensors, actuators, controllers).
- Centralized ECS Controller Unit manages all ECS functions and interfaces with other aircraft systems (engine bleed air, ice protection, pressurization, oxygen).
- Redundancy is achieved at the system level (e.g., dual ECS packs, cross-bleed valves), but controller-level redund. and communic. protocols are not specified.
- Subsystems (air cycle machine, heat exchang., water separators, control valves) are physically and functionally integrated to provide stable, safe, and efficient environmental control.

Requirements Documentation

The following tables are extracted from the original ECS synthesis report. The acceptance criteria column was removed for readability.

Table AT.10.: ECS synthesis report - excerpt of the performance requirements list

ID	Title	Description	Verification Method	Transparency
NFR. P.001	Temperat. Control Range	The ECS shall maintain cabin supply air temperature between 1.7°C (35°F) and 24.4°C (76°F), with setpoint control accuracy of ±2°C and zone to zone variation <2°C.	Environment. chamber test, flight test, simulation	RQ.4,FU.1.1,RQ.1.1.2 [Setpoint accuracy/ zone variation assumed; 8th step]
NFR. P.002	Humidity Control	The ECS shall maintain cabin relative humidity within 10-60% RH range, with water separator efficiency >85% and condenser outlet moisture at least 20°F below dew point.	Humidity test, water separator test	FU.1.1, SC.2.2.1.2 [RH range assumed]
NFR. P.003	Pressure Regulation	The ECS shall provide cabin pressure by maintaining a differential of up to 8.3 psi, with system pressure during cruise at 30-45 psig and overpressure protection at 55 psig.	Pressure test, simulation	SC.2, FU.1.1
NFR. P.004	Response and Settling Time	The ECS shall achieve setpoint temperature within 5 minutes of a step change in demand under nominal cond.	Step response test, simulation	[Assumed: Industry standard; not specified in Q&A]
NFR. P.005	Thermal Load Handling	The ECS shall provide a total cooling capacity of at least 50 kW per aircraft (typical for commercial narrowbody), with supply air mass flow rates of ≈ 1.8 kg/s per pack.	Load test, simulation	[FU.1.1, SC.5.1]

Table AT.11.: ECS synthesis report - excerpt of the functional requirements list

ID	Title	Description	Prio.	Verification Method	Transparency
FR. 001	Cabin Temp. Control	The ECS shall provide temp.-controlled air to the flight deck, passenger cabin, and cargo compartments, maintaining setpoint temp. between 18-24°C under all operational conditions.	Must Have	System level test, simulation, and flight test	FU.1, FU.1.1, SC.3.1
FR. 002	Cabin Pressuriz. Control	The ECS shall maintain cabin pressure equivalent to an altitude of 8,000 ft (2,440 m) with a maximum pressure differential of 8.3 psi, throughout all flight phases.	Must Have	Pressure sensor data review, simulation, and flight test	FU.1, FU.1.1, SC.3.1
FR. 003	Cabin Humidity Control	The ECS shall provide both automatic temperature and moisture from supply air, maintaining water separator efficiency above 85%, and preventing condensation or ice formation in the cabin and ECS components.	Must Have	Humidity sensor data, water separator test, environmental chamber test	FU.1.1, SC.3.2.1.2
FR. 004	Vent. & Air Quality	The ECS shall supply fresh air/outside air to cabin/occupied compartments, maintaining air quality by filtering contaminants and providing sufficient ventilation per regulatory requirements.	Must Have	Air qual. test, filter performance test, regulatory compliance review	FU.1, FU.3.2.1, SC.3.2.2
FR. 005	Cabin Air Distribution	The ECS shall provide distributed air uniformly to all cabin zones, minimizing temperature gradients and ensuring zone-specific temperature control via mixing units and temperature control valves.	Should Have	Zone temp. mapping, air flow measurement, simulation	RQ.1.1.2, FU.1.1, FU.1.2
FR. 006	Autom. & Manual Control Modes	The ECS shall provide both autom. and manual control modes for temperature, pressure, and ventilation, with flight crew able to override automatic settings as needed.	Must Have	Flight deck interface test, funct. demonstration	SC.3, FU.3, FU.1.1
FR. 007	Emerg. & Backup Operation	The ECS shall provide emergency ventilation and pressurization via redundant packs, cross-bleed valves, APU bleed air, and emergency ram air in the event of primary system failure.	Must Have	Failure mode simulation, emergency procedure test	[FU.3, FU.4.2, AP.3, SC.2.2]
FR. 008	Health Monit. & Fault Detection	The ECS shall continuously monitor system health using sensors (temperature, pressure, humidity, airflow, valve position, turbine speed) and provide early fault detection, isolation, and reporting to maintenance personnel.	Must Have	BIT test, ACMS data review, simulated fault injection	[FU.5, FU.5.1, SC.3.1, SC.3.1.2]
FR. 009	Condition-Based Maint. Support	The ECS shall support condition-based maintenance by providing prognostic health indicators, remaining useful life estimates, and maintenance alerts based on sensor data and diagnostic algorithms.	Should Have	Maint. log review, health index validation, maintenance scheduling test	[FU.5, SC.3.1, FU.5.2]
FR. 010	Integrat. with Aircraft Systems	The ECS shall interface with engine/APU bleed air, ram air, electrical power, pressurization, avionics, and water drainage systems, exchanging required signals and maintaining operational coordination.	Must Have	Integration test, interface simulation, system-level test	[SC.1, SC.5, LE.3, LE.5]

Table AT.12.: Continuation of the functional requirements list

ID	Title	Description	Priority	Verification Method	Transparency
FR. 011	Support for All Flight Phases	The ECS shall operate effectively during ground, climb, cruise, descent, and emergency flight phases, automatically adjusting control strategies to maintain environmental conditions.	Must Have	Flight simulation, environmental chamber test, flight test	[FU.3, SC.2, SC.2.2, SC.5.1.2]
FR. 012	Anti-Icing & Moisture Managem.	The ECS shall prevent ice formation in critical components by controlling turbine discharge temperature, using anti-icing airflows, and managing water extraction and reheating as required.	Must Have	Cold soak test, ice formation test, sensor data review	[RQ.4, FU.1.1, SC.2.2, SC.2.2.2]
FR. 013	Contamin. Filtration and Air Quality	The ECS shall integrate HEPA filters and particulate retention media in recirculation air paths to remove contaminants, with filter maintenance intervals defined by environmental conditions and system monitoring.	Should Have	Filter efficiency test, maintenance record review	[SC.3.2.2, FU.3.2.2]

Functions Documentation

Table AT.13 presents the functional hierarchy and classification extracted from the comprehensive ECS report. This table establishes the primary functional breakdown of the ECS, with the main function (FN.001) serving as the root from which all primary functions are derived as child functions. The complete functions documentation contains extensive additional information, including detailed functional flow representations with sequential flow tables and state transition tables, comprehensive use cases covering both nominal and off-nominal operational scenarios, detailed functional specifications with timing and interface specifications, control algorithms and decision logic tables, as well as complete cross-reference mappings between functions, requirements, and system context elements.

Table AT.13.: ECS synthesis report - function hierarchy and classification

Func ID	Name	Classif.	Parent/ Child	Source
FN.001	Cabin Environmental Control	Main	Root	FU.1
FN.002	Temperature Control	Primary	Child of FN.001	FU.1.1, FU.1.1.1
FN.003	Pressurization Control	Primary	Child of FN.001	FU.1.1
FN.004	Humidity and Moisture Control	Primary	Child of FN.001	FU.1.1, FU.1.1.1
FN.005	Ventilation and Air Quality	Primary	Child of FN.001	FU.1, FU.3.2.1, SC.3.2.2
FN.006	Cabin Air Distribution	Primary	Child of FN.001	FU.2, RQ.1.1.2
FN.007	System Monitoring and Diagnostics	Primary	Child of FN.001	FU.2, FU.5, FU.5.1
etc.				

Table AT.14.: Main ECS Functions (excerpt, simplified)

ID	Description	Input Parameters	Output Parameters
FN-001	Maintains safe, comfortable, and habitable environment for crew, passengers, and equipment by controlling temperature, pressure, humidity, and air quality in all occupied compartments throughout all flight phases.	• Bleed air T, P, flow • Ram air temperature/flow • Flight phase/mode • Crew inputs • Cabin T, P, humid. set-points • Sensor data (T, P, humid., flow, valve positions)	• Air quality metrics • Fault/health status • Maintenance alerts • Conditioned air (T, P, humid.) to cabin/ cockp./cargo
FN-002	Regulates the temperature of air supplied to all zones by modulating bleed air flow, mixing with recirculated and trim air, and controlling heat exchangers and turbines to maintain setpoints.	• Bleed air T/P/flow • Ram air T/flow • Cabin temp. set-points • Sensor data (T, P, flow)	• Conditioned air temp. • Valve positions • Fault/health status
FN-003	Maintains cabin pressure within safe limits by regulating air supply and controlling outflow valves, ensuring compliance with altitude and structural limits.	• Bleed air P/flow • Cabin pressure setpoint • Sensor data (P, altitude)	• Cabin pressure • Outflow valve position • Fault/health status
FN-004	Removes moisture from supply air using water separators, condensers, and reheaters to prevent condensation, ice formation, and maintain passenger comfort.	• Bleed air humidity • Cabin humidity setpoint • Sensor data (humid., T, P)	• Conditioned air humidity • Water drain • Fault/health status
FN-005	Supplies fresh and recirculated air, filters contaminants, and maintains air quality and ventilation rates per regulatory requirements.	• Bleed air flow • Recirculation air • Sensor data (air quality, flow, filter status)	• Cabin air quality • Filter status • Fault/health status
FN-006	Distributes conditioned air uniformly to all cabin zones, using ducting, mixing manifolds, and zone temperature control valves to minimize gradients and ensure comfort.	• Cond. air (T, P, humidity) • Zone setpoints • Sensor data (zone T, flow)	• Zone airflows • Zone temperatures • Fault/health status
FN-007	Continuously monitors ECS health using sensors, detects faults, extracts health indicators, and provides diagnostic/prognostic information for maintenance and crew.	• ACMS data • Sensor data (T, P, humid., flow, valve positions, turbine speed)	• Health indicators • Fault reports • Maintenance alerts
etc.			

Logical Elements Documentation

Table AT.15 presents the logical component hierarchy extracted from the ECS logical elements and architecture documentation. The table identifies nine primary logical subsystems that collectively implement the Environmental Control System functionality, ranging

from core air conditioning components to auxiliary systems for monitoring and redundancy management. The complete logical elements documentation provides extensive additional architectural information including logical component relationship mappings that specify dependencies and service provision patterns between subsystems, state models covering operational states from ground operation through emergency modes, and a logical architecture organized into five layers (System Integration, Control & Monitoring, Environmental Conditioning, Distribution & Cabin Interface, and Redundancy & Emergency). It also includes information flow specifications with data exchange patterns, timing requirements, and communication protocols, architectural patterns and design principles such as centralized control architecture and condition-based maintenance approaches, as well as traceability matrices linking logical elements to functional requirements.

Table AT.15.: ECS synthesis report - logical component hierarchy and classification

ID	Name	Description & Purpose	Related Physical Components	Source
LE. 001	Air Cycle Machine (ACM) Subsystem	Core subsystem for cooling and conditioning hot engine bleed air by compressing, expanding, and exchanging heat with ram air.	Compressor, turbines, primary/secondary heat exchangers, fan	LE.1, PC.1, PC.2
LE. 002	Water Separation Subsystem	Removes moisture from conditioned air to prevent condensation and icing; drains condensate.	Water extractor, condenser, reheater, drain lines	LE.1, FU.1.1, SC.4.2.1.2, RQ.5.1.2
LE. 003	Heat Exchang. Subsystem	Transfers heat between hot bleed air and cooling ram air to reduce air temperature efficiently.	Primary, second., regenerative, condenser, reheater	LE.1, PC.1, FU.1.1
LE. 004	Control Subsystem	Centralized logic for monitoring, regulating, and coordinating ECS operation using sensor interfaces.	Controller hardware, actuators, sensors	LE.1, LE.3, FU.1.1.1, FU.2, FU.5
LE. 005	Auxiliary Subsystems	Enhances efficiency, filtration, and operational flexibility, includes recirculation, filtration, and evaporative cooling.	Fans, filters, water injectors	LE.1, FU.1.1, FU.3.2.2, LE.5.1.2
LE. 006	Mixing and Distribution Subsystem	Mixes conditioned and recirculated air, distributes to zones, and manages zone-specific temperature control.	Mixing unit, ducting, zone valves	FU.1.2, RQ.1.1.2
LE. 007	Pressuriz. & Outflow Subsystem	Maintains cabin pressure by regulating air outflow and integrating with CPCS.	Outflow valve, sensors, CPCS interface	FU.1.1, SC.1, FU.4.2.1
LE. 008	Health Monit. and Diagnosis Subsystem	Monitors ECS health, detects faults, extracts health indicators, and supports condition-based maintenance.	ACMS, sensors, controller	FU.5, FU.5.1, SC.3.1, SC.3.1.2
LE. 009	Redundancy & Emergency Subsystem	Ensures continued ECS operation during component failures by managing secondary air sources.	Redundant packs, cross- bleed, emergency ram air	FU.4.2, AP.3, SC.2.2

Components Documentation

The physical components of the ECS have been systematically documented and analyzed, with Table AT.16 presenting the core hardware and software components that implement the ECS functionality.

Table AT.16.: ECS synthesis report - physical component hierarchy and classification

ID CMP-	Name	Description	Parent	Trace. Log. El.	Transp.
HW.001	Air Cycle Machine (ACM)	Core air cond. unit containing compressor, turbines, and fan, providing cooling and pressure regulation via air cycle.	System	LE-001	PC.1; LE.1
HW.002	Compressor	Compresses ram air or bleed air to increase pressure and temperature for cooling cycle.	HW.001	LE-001	PC.1; LE.1
HW.003	Turbine (Primary/Second.)	Expands and cools air, extracting energy to drive compressor/fan; limits discharge temperature to prevent ice.	HW.001	LE-001	PC.1; LE.1
HW.004	Fan	Drives ram air through heat exchangers, especially during ground/low speed operation.	HW.001	LE-001	PC.1; LE.1
HW.005	Primary HX (PHX)	Cools hot engine bleed air using ram air, first stage of air cooling.	System	LE-003	PC.1; LE.1
HW.006	Secondary HX (SHX)	Further cools process air before expansion in turbines.	System	LE-003	PC.1; LE.1
HW.007	Regenerative HX (RGX)	Recovers heat between process streams to improve cycle efficiency.	System	LE-003	PC.1; LE.1
HW.008	Condenser	Cools air below dew point for moisture removal; maintains temperature above freezing to prevent ice.	System	LE-003	PC.1; LE.1
HW.009	Reheater	Recovers energy and reheats air after condensation to prevent ice form.	System	LE-003	PC.1; LE.1
HW.010	Water Extractor	Removes condensed water from cooled air; efficiency >85%.	System	LE-002	PC.1; LE.1
HW.011	Drainage System	Heated drain lines remove water from ECS to aircraft drainage.	System	LE-002	SC.5; LE.1
SW.001	ECS Controller Unit	Centralized logic for ECS operation, monitoring, diagnostics, and interfaces with aircraft systems.	System	LE-004	LE.1; LE.3
HW.012	Flow Control Valve (FCV)	Regulates mass flow of engine bleed air into ECS packs; critical for pressure and flow control.	System	LE-004	PC.4; LE.1
HW.013	Temperature Control Valve (TCV)	Modulates temperature of conditioned air by mixing or bypassing air streams; maintains cabin setpoint.	System	LE-004	PC.4; LE.1
HW.014	Bypass Contr. Valve (BCV)	Modulates flow around turbine/heat exchanger to fine-tune temp. & press.	System	LE-004	PC.4; LE.1
HW.015	Check Valves (CV1, CV2, PCK-V, EDV, CSV, TACVV, TACV)	Prevent backflow and isolate ECS sections for redundancy and failover.	System	LE-004, LE-009	PC.4; LE.1; AP.3.2

Table AT.17.: ECS synthesis report - physical component hierarchy and classification (Part 2)

ID CMP-	Name	Description	Parent	Trace. to Log. El.	Trans-parency
HW.016	Ram Air Regulating Valve (RARV)/ Air Vapor Valve (AVV)	Controls ram air flow through heat exchangers for thermal management.	System	LE-004	PC.4; LE.1
HW.017	Condensing Loop Isolation Valve (CLIV)	Isolates condensing loop for maintenance or operational flexibility.	System	LE-004	PC.4; LE.1
HW.018	Secondary System Temperature Bypass Valve (SSTBV)	Fine-tunes temperature control by modulating bypass flow in advanced cycles.	System	LE-004	PC.4; LE.1
HW.019	Recirculation Fan	Draws cabin air through HEPA filters for recirculation and energy efficiency.	System	LE-005	PC.1; LE.1
HW.020	HEPA Filter	Removes particulates and contaminants from recirculated air; efficiency >99% for 0.3 micron particles.	System	LE-005	SC.3.2.2; FU.3.2.2
HW.021	Ram Air Spray Injector	Sprays extracted water into ram air for evaporative cooling and improved heat exchange performance.	System	LE-005	LE.5.1.2; PC.1
HW.022	Mixing Manifold	Combines conditioned air from packs with recirculated air and distributes to zones.	System	LE-006	PC.1; 1.2; FU.1.2

Architectural Views

In addition to the requirements and interface views, the following excerpts illustrate how the LLM-based synthesis successfully generated the *Functional Architecture View* and the *Physical View* of the ECS. The tables provide representative sections of both perspectives as generated within the knowledge synthesis report. They demonstrate that the model was able to correctly identify hierarchical functional structures and allocate relevant components. The *Functional Architecture View* focuses on the decomposition of system functions and their assigned components, while the *Physical View* presents the corresponding hardware allocation at component level.

Table AT.18.: Excerpt of ECS Functional Decomposition (simplified)

Function ID	Sub-functions	Allocated Components	Trans-parency
AV-F-001	Temperature, Pressurization, Humidity, Air Quality, Distribution, Monitoring, Transition, Safety, Redundancy, Integration	ACM, Heat Exchangers, Water Separator, Valves, Sensors, Controller	Req. FR-001 to FR-014
AV-F-002	Bleed Air Regulation, Ram Air Regulation, Mixing, Zone Control	FCV, TCV, TAV, Sensors	FU.1.1, FU.1.1.1, RQ.4
AV-F-003	Outflow Regulation, Pressure Sensing	Outflow Valve, Pressure Sensors	FU.1.1, SC.1
AV-F-004	Condensation, Water Extraction, Reheating, Drainage	Water Separator, Condenser, Reheater, Drainage	FU.1.1, SC.4.2.1.2
AV-F-005	Filtration, Recirculation, Air Quality Sensing	Recirculation Fan, HEPA Filter, Sensors	FU.3.2.2, SC.3.2.2
AV-F-006	Health Monitoring, Fault Detection, Prognostics	ACMS, Sensors, Controller	FU.5, FU.5.1, SC.3.1
AV-F-007	Mode Sequencing, Emergency Handling	Controller, Valves	FU.3, FU.4.2
AV-F-008	Threshold Enforcement, Emergency Ventilation, Anti-Icing	Controller, Emergency Valves, Redundant Paths	FU.4, RQ.4
AV-F-009	External System Coordination, Interface Management	Controller, Interface Modules	LE.1, LE.2, LE.3

Table AT.19.: Excerpt of ECS Physical Architecture View (simplified)

Comp. ID	Component Name	Physical Location/Zone	Physical Interfaces	Trans-parency
AV-P-001	Air Cycle Machine (ACM)	ECS bay (fuselage)	Bleed air duct, ram air duct, electrical, sensors	PG.1, LE.1
AV-P-002	Primary Heat Exchanger (PHX)	ECS bay, ram air duct	Bleed air, ram air, mounting flanges	PG.1, LE.1
AV-P-003	Water Extractor	Downstream of turbine or condenser	Air duct, drain line	PG.1, LE.1
AV-P-004	Flow Control Valve (FCV)	Upstream of ACM, bleed air duct	Bleed air duct, controller wiring	PC.4, LE.1
AV-P-005	Temperature Control Valve (TCV)	Downstream of ACM, conditioned air duct	Conditioned air duct, controller wiring	PC.4, LE.1
AV-P-006	Mixing Manifold	Cabin supply ducting	Conditioned air, recirculation air, zone ducts	RQ.1.1.2, FU.1.2
AV-P-007	Recirculation Fan & HEPA Filter	Recirc. duct, cabin	Recirc. duct, filter housing, electrical	SC.3.2.2, FU.3.2.2
AV-P-008	Outflow Valve	Pressure vessel boundary	Cabin air, external vent, controller wiring	LE.007, FU.1.1

A.11. Evaluation of the ECS Knowledge Synthesis Report

Source Fidelity and Synthesis Reliability

A systematic evaluation of the source fidelity was conducted to assess the reliability of LLM-assisted knowledge synthesis in maintaining traceability to source materials rather than generating hallucinated content. This evaluation addresses a critical concern in AI-assisted engineering documentation: whether synthesized statements can be considered trustworthy representations of the underlying knowledge base.

1. Methodology and Scope
The evaluation method focused on green-classified statements (n = 183) that were explicitly marked as "directly traceable to source content." Each technical claim from the synthesized ECS documentation was systematically verified against its Q&A source pair in order to assess the accuracy of the knowledge transfer and the fidelity of the source during synthesis. This was a statement-by-statement assessment.

2. Assessment Results
The evaluation shows a high fidelity between synthesized content and source material, with 168 statements (91.8 %) demonstrating direct source agreement, see Table AT.20. These source-matching statements represent technical claims that can be traced directly to their corresponding QA-pairs without modification or interpretation, indicating high-quality knowledge transfer during the synthesis process. The predominance of source-matching assertions provides strong evidence for the reliability of the LLM-assisted synthesis methodology in maintaining technical accuracy.

Table AT.20.: Source fidelity distribution of synthesized statements

Correctness Level	Count	Percentage	Interpretation
Source-Aligned	168	91.8 %	Statement directly traceable to source content
Value-Added	13	7.1 %	Statement correct but includes values/details, which are not in source
Source-Divergent	2	1.1 %	Statement not traceable to or contradicts source

Thirteen statements (7.1%) were value-added, containing specific values or technical details not present in the original source content. For example, the statement about the cabin pressure difference of "8-9 psi" was not present in the referenced source. However, further analysis revealed that some of these values or statements, including the 8-9 psi specification, were present in at least two other sources within the corpus, but had not been attributed to them. These statements represent cases in which the synthesis process either generated additional technical specifications that went beyond what was explicitly stated in the source material or drew on correct information from the corpus without referencing the corresponding sources. This distinction is important because it differentiates between fabricated information and existing, but incorrectly attributed, data. The moderate percentage of value-altered statements indicates cases where the synthesis exceeded the limits of the source material, necessitating careful review for technical applications where precision and traceability are paramount.

The evaluation identified only two source-derogating statements (1.1 %) that could not be traced back to or contradicted their referenced source material. One notable example included a safety requirement statement that asserted, "The ECS shall be designed to meet safety requirements for catastrophic and hazardous failure modes as defined by applicable aviation regulations (e.g., FAA/EASA)." When examining the referenced source QA-pair, this regulatory compliance requirement was not explicitly stated in the source material. It is important to note that these statements were not technically false per se, but rather represented technically plausible assertions that were not substantiated in the sources. The critical problem lies in the green classification of these statements: if the LLM had marked them as orange or red, it would have been immediately apparent to engineers that additional verification was required. However, the green marking falsely suggests direct source support and can lead to misinterpretation in safety-critical applications.

3. Source Coverage and Knowledge Distribution
The knowledge synthesis report produced by the LLM is based on 48 QA pairs out of a possible 183, as shown in Figure AF.4. The selection pattern reveals strategic source utilization: System Context (SC) demonstrated the highest utilization rate with 17 QA-pairs used from 31 available (55 %), followed by Functions (FU) with 13 pairs from 29 available (45 %), while Logical Elements (LE) and Physical Components (PC) each contributed 6 pairs from 20 and 31 available respectively (30 % and 19 %). Requirements (RQ) provided 4 pairs from 32 available (13 %), and Architectural Patterns (AP) contributed 2 pairs from 22 available (9 %). This selective utilization demonstrates the LLM's ability to identify and extract the most relevant information from each knowledge domain while maintaining comprehensive coverage across all critical areas necessary for ECS understanding.

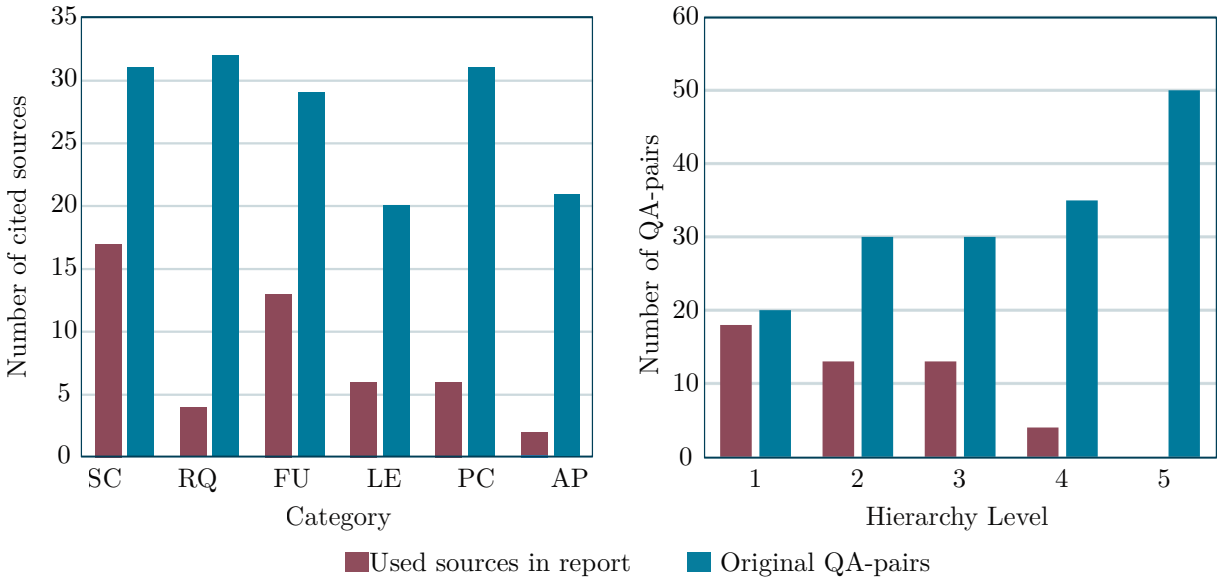

Figure AF.4.: Source utilization analysis showing LLM selection from available QA-pairs across knowledge categories and hierarchical levels, demonstrating strategic information extraction for comprehensive synthesis

The hierarchical analysis reveals knowledge extraction spanning five distinct levels of system abstraction, with the LLM utilizing varying proportions from each level. Level 1 contributed 18 QA-pairs from 20 available (90 %), Level 2 provided 13 QA-pairs from 30 available (43 %), Level 3 contributed 13 QA-pairs from 31 available (42 %), Level 4 provided 4 QA-pairs from 35 available (11 %), and Level 5 contributed 0 QA-pair from 51 available. This hierarchical distribution pattern indicates that the LLM strategically focused on higher-level abstractions (Level 1) and mid-level technical details (Levels 2 and

3) while selectively incorporating specific implementation details from lower levels. The resulting knowledge representation ensures balanced coverage across multiple abstraction levels, which are necessary for engineering design applications, demonstrating the synthesis methodology's capability to extract relevant information across the entire system hierarchy.

4. Conclusion
The source fidelity evaluation demonstrates that the LLM-based knowledge synthesis is highly reliable for applications in aerospace engineering. With 91.8 % of statements directly traceable to their sources and only 1.1 % showing divergence, the synthesis process maintains strong technical accuracy.

The main risk identified is not the generation of incorrect information, but rather the misclassification of confidence levels. The 2 source-divergent statements were technically reasonable but incorrectly marked as green, suggesting direct source support where none existed. This misclassification could mislead engineers in safety-critical applications, as they might rely on information that appears source-verified but actually requires additional validation.

The evaluation covered 48 QA-pairs strategically selected from a larger knowledge base, with particularly strong utilization of high-level system concepts (85 % at Level 1) and balanced coverage across technical domains. This comprehensive scope ensures the results represent the entire synthesis methodology.

Future work should focus on automating the statement-by-statement evaluation process to enable scalable evaluation of larger knowledge bases. Similar to the automated evaluation approach demonstrated in Study C (Section 4.2.3), this could involve a Python script for extracting statements and mapping source references, followed by LLM-based comparison between synthesized statements and their claimed sources. An additional evaluation loop could then reassess and potentially adjust the green classification based on the automated analysis, reducing the risk of confidence misclassifications while maintaining evaluation consistency across larger datasets.

Overall, LLM-assisted synthesis shows strong potential for aerospace documentation, provided that confidence classification systems are properly validated and verification workflows are implemented to address the identified classification risks.

Confidence Classification Assessment

Following the source fidelity evaluation, a comprehensive assessment of the confidence classification system was conducted to evaluate the effectiveness of the three-tier visual coding scheme in communicating information reliability and identifying knowledge gaps within the synthesized ECS documentation.

1. Classification System Design and Purpose
The confidence classification employs a color-coded system to distinguish between different levels of information certainty: Green indicates direct source traceability, orange represents derived or combined information from multiple sources, and red highlights assumptions or identifies explicit knowledge gaps. This visual classification system enables engineers to rapidly assess the reliability of technical information and identify areas requiring additional verification or specification.

The assessment examined the relationship between confidence classifications and actual source fidelity, particularly focusing on the performance of green-classified statements and the practical utility of red classifications in identifying critical knowledge gaps for engineering applications.

2. Green Classification Performance
The evaluation of green-classified statements demonstrates exceptional reliability, with over 90 % of statements marked as "directly traceable to source content" achieving verified source alignment. This high accuracy rate confirms that green classifications serve as reliable indicators of source-grounded information, enabling engineers to distinguish between verified technical data and synthesized assertions with high confidence.

The strong performance of green classifications establishes a foundation of trustworthy technical information within the synthesized documentation, supporting informed engineering decision-making during preliminary design phases where source-verified data is critical for system architecture development.

3. Red Classification Effectiveness
The red classification system demonstrates particular value in explicitly identifying critical knowledge gaps and specification limitations within the source documentation. Representative examples of red-classified knowledge gaps include:

- "Detailed material selection, actuator types, and communication protocol specifications are not available in the provided documents. These are highlighted as critical gaps for further clarification."
- "Detailed interface mechanisms (e.g., data bus protocols) and cross-layer security/iso-lation boundaries are not specified in the provided documents."
- "Detailed user interface flows and abnormal mode transitions for all possible ECS failures are not fully specified in the provided documents."

These red classifications serve as explicit risk indicators, highlighting areas where assumptions have been made or where critical technical specifications are absent from the knowledge base. This transparent identification of knowledge limitations enables engineers to prioritize additional research and specification development activities.

4. Classification System Assessment
Despite the overall effectiveness of the classification system, some inconsistencies were observed. Statements containing acknowledged knowledge gaps or assumptions were sometimes classified as green instead of orange or red. This indicates opportunities for improving classification accuracy, especially when distinguishing directly traceable information from statements with embedded uncertainties.

However, the systematic presentation of information uncertainties through red classifications provides significant engineering value. Red classifications highlight knowledge gaps, enabling risk-informed design decisions. They also help identify areas requiring additional technical investigation or specification development.

The confidence classification system functions as an effective engineering risk management tool. It provides an immediate visual indication of information reliability and specification completeness. Red classifications enable proactive identification of technical risks and support a systematic approach to addressing specification deficiencies in aerospace systems.

Traceability Assessment

The traceability assessment employed a systematic approach to evaluate traceability across multiple architectural abstraction layers. Established systems engineering standards such as ISO/IEC/IEEE 15288 [11] and INCOSE Systems Engineering Handbook [12] mandate that system elements be traceable to each other. The methodology tests

whether this traceability requirement is fulfilled in LLM-generated content and evaluates the quality of these traced relationships.

1. Assessment Framework and Methodology
The assessment framework employs a hierarchical traceability model spanning the three development steps - requirements, functions, logical elements, and components. Each traceability relationship between two elements was evaluated using a three-state classification system:

1. Correct: Mapping exists and is semantically valid
2. Incorrect: Mapping exists but is semantically invalid
3. Missing: Expected mapping is absent

The framework employs several quantitative metrics to provide an objective assessment of traceability quality. For each traceability layer, the accuracy metric was calculated as the ratio of correct mappings to total mappings. The completeness ratio quantifies the proportion of expected traceability relationships that are present.

2. Comprehensive Assessment Results
The framework was applied to assess the knowledge synthesis report, comprising 47 individual traceability mappings across three architectural layers, as shown in Table AT.21.

Table AT.21.: Overall traceability assessment results

Traceability Category	**Total Mappings**	**Correct Mappings**	**Incorrect Mappings**	**Missing Mappings**	**Accuracy (%)**
Requirements → Functions	22	22	0	0	100
Functions → Logical Elements	13	13	0	0	100
Logical Elements → Components	12	12	0	0	100
TOTAL	**47**	**47**	**0**	**0**	**100**

The assessment reveals that the traceability quality is 100 % accurate across all layers. This perfect score indicates that all expected traceability relationships were correctly identified and semantically validated in the LLM-generated documentation.

Traceability Quality Analysis and Implications

The perfect traceability score of 100 % represents a significant achievement for automated documentation generation, with several important implications:

- Methodological Rigor: The complete absence of incorrect or missing mappings demonstrates that the LLM successfully maintained logical consistency across all architectural abstraction layers. Each requirement was correctly traced to its implementing functions, which were accurately mapped to logical elements and subsequently to physical components. This end-to-end traceability integrity is essential for systems engineering verification and validation activities.
- Semantic Consistency: The 100 % correctness rate indicates that the LLM not only identified structural relationships but also maintained semantic validity across mappings. For instance, temperature control requirements were correctly traced to thermal management functions and subsequently to heat exchanger components, demonstrating deep understanding of domain-specific relationships.

- Documentation Quality Standard: This result establishes a high benchmark for LLM-generated systems documentation quality. The perfect traceability score suggests that automated documentation generation can meet or exceed manual documentation standards, which often suffer from incomplete or inconsistent traceability links [222].
- Engineering Process Integration: The comprehensive traceability enables seamless integration into established systems engineering processes, supporting change impact analysis, verification planning, and requirements validation activities. The bidirectional traceability facilitates both forward tracking (requirements to implementation) and backward tracking (components to originating requirements).
- Scalability Considerations: The systematic achievement of perfect traceability across 47 mappings demonstrates the scalability potential of LLM-based documentation generation. This suggests that larger, more complex systems could benefit from similar automated approaches while maintaining traceability integrity.

Traceability underscores the suitability of LLM-generated documentation for complicated technical systems and builds confidence in the reliability of automated knowledge synthesis for applications in the field of systems engineering.

Conclusion and Methodology Evaluation

The comprehensive evaluation confirms that LLM-assisted knowledge synthesis can achieve engineering-grade accuracy suitable for aerospace system development. The methodology demonstrates the capability to maintain technical fidelity, provide effective risk communication, and ensure architectural coherence while significantly reducing manual effort requirements. The systematic assessment offers a replicable methodology for evaluating the quality of AI-assisted knowledge synthesis in technical domains. The evaluated quality metrics establish benchmarks for future developments in automated documentation generation, providing confidence for practical engineering applications. The results indicate that the knowledge synthesis methodology represents a viable approach for transforming systems engineering practice through intelligent automation, while maintaining the rigorous quality standards required for complex aerospace systems development.

A.12. Representative Excerpts from ECS Architecture Generation Report

The comprehensive architecture generation process produced detailed technical documentation. To demonstrate the quality of the generated content, representative excerpts from key sections are presented below. The following tables represent a small subset of the complete documentation and serve to exemplify the structured analysis, comprehensive coverage, and consistent application of the four-tier classification system throughout the architecture generation process.

All text, lists, and highlighted color codes shown in the following excerpt are *directly and unmodifiedly taken from the automatically generated report.* The formatting, structure, and color annotations have been preserved exactly as produced by the LLM during the architecture generation process.

Note: All content directly reproduced from the original LLM-generated report is highlighted in light gray. This convention applies uniformly to text passages, lists, and tables.

Executive Summary from Report

This document analyzes reusability of components from the existing Bleed-Air-based ECS (Section 5, ECS Components Documentation) when transitioning to an Ambient Air Intake ECS architecture. The new system will draw in outside air from the aircraft fore region at approximately 0.2-0.3 bar and temperatures of about -60°C to +50°C (derived from system context: external temperature range, SC.2.2, ambient intake approach).

- Key Findings: Many core elements (e.g., distribution ducts, certain valves) can be reused with modifications, but the compression stage requires new or heavily modified components (due to lower inlet pressure and different temperature profile).
- Critical New Components: A dedicated ambient air compressor and associated heat exchangers sized for lower inlet pressures will be required (analysis specifically for 0.2-0.3 bar intake).
- Reusability: High for ducting, sensors/monitoring systems; Medium for control valves (require recalibration); Low for existing bleed-air-based turbines and compressors (since they are optimized for 30-45 psig bleed air).
- Overall Feasibility: The ambient intake ECS is feasible, but requires major modification in compression stages (dedicated new compressor for 0.75 bar cabin pressure target) and revised cooling/anti-icing strategies.

Component Reusability Matrix

The following table summarizes the component-level reusability assessment as generated in the report. It illustrates how existing ECS components were evaluated for applicability within the newly synthesized ambient air intake architecture.

Table AT.22.: ECS generation report - component reusability matrix

Comp. ID	Comp. Name	Current Function	Current Oper. Cond.	New Operating Cond.	Reusability Status	Required Modifications	Performance Impact
CMP-HW-001	Air Cycle Machine (ACM)	Core air cond. using bleed air expansion	30-45 psig, 180-250°C inlet	0.2-0.3 bar, -60 to +50°C inlet from ambient	MAJOR Modification	Replace or redesign compressor/turbine stages to handle lower pressure and different temperat.	Reduced efficiency if used "as-is." Optimized design needed
CMP-HW-012	Flow Control Valve (FCV)	Regulates mass flow of bleed air	30-45 psig bleed air	Amb. pressure ∼0.2-0.3 bar; must regulate compres. air downstream ∼0.75 bar	MINOR Adapation	Valve trim/actuator settings updated for new pressure ranges	Slightly different flow curve; minimal perf. penalty
CMP-HW-022	Mixing Manifold	Combines cond. air with recirculated air	∼1.8 kg/s total airflow, 18-24°C cabin supply	Same flow rate target but source pressure at 0.75 bar post-compression	DIRECT Reuse	No modification needed (manifold design suitable for new supply conditions)	Minimal or no impact on perf.
CMP-HW-010	Water Extractor	Removes moisture; >85% efficiency	177-232°C bleed air, post-cooling	Possible lower temp. after compression (∼45°C at cruise compr.)	MINOR Adapation	Recheck capacity for lower inlet temperature; may need minor resizing or new coalescer element	Similar function, slight perf. shift if lower humid./temp.
CMP-SW-001	ECS Controller Unit	Controls ECS modes, monitors sensors	Interfaces with bleed air operating logic	New logic to manage ambient intake compression stage	MAJOR Modification (Software)	New control algorithms for compressor speed, surge margin with ambient intake airflow	Perf. depends on updated code integration; no hardware limitations expected
CMP-HW-025	Outflow Valve	Regulates cabin pressure via CPCS	Up to 8 - 9 psi cabin differential	Same pressuriz. target (0.75 bar cabin vs. ∼0.25 bar external at altitude)	DIRECT Reuse	No major hardware change required; same func- tion	No perf. penalty; outflow operation remains as-is
CMP-HW-031	Elect. Driven Compressor (MEA opt.)	Provides compr. air for ECS	Limited or no use in traditional bleed-air environment	Critical function to compress 0.2-0.3 bar ambient to ∼0.75 bar	REPLAC NEW COMPONENT	Full sizing for ambient intake flow, HPC design for 3-4x compression ratio depends on flight altitude	Enables stable supply but requires robust power electronics
etc.							

Ambient Air Intake ECS: Complete Requirements, Functions, and Logical Design

Table AT.23.: ECS generation report - external system boundary definition

External System	System Type	Interaction Description	Interface Requirements	Dependency Level	Change from Original ECS
Fore Fuselage Ambient Inlet	Ambient Air Source	Supplies ∼0.2 - 0.3 bar external air for ECS compression	Must accommodate icing protection, intake sensor da- ta	High	NEW (Ambient intake replaced bleed source)
Engine/APU Bleed System	Pneumatic Source	Now used as optional or backup source if dual-mode is retained	Reduced or standby interface if ambient compression is primary	Low (in Ambient-only config.)	MODIFIED (was primary, now second./ backup)
Ram Air System	Cooling Air Source	Cool ECS heat exchangers with ambient flow at variable altitude	Valves, fans remain as before for heat rejection	Medium	UNCHANGED
Aircraft Electrical System	Electrical Power	Supplies power to the newly introduced HPC (High-Pressure Compressor) or electric compressor unit	Must handle HPC start-up loads plus steady-state ∼10 - 15 kW (approx.)	High for HPC-driven ECS	NEW or significantly expanded draw
Cabin Pressurization System	Pneumatic/ Mechanical	Regulates cabin outflow; interacts with ECS to maintain 0.75 bar cabin supply at altitude	Outflow valve logic unchanged	High	UNCHANGED
Water Drainage System	Drain Lines	Removes moisture from separator to external drain	Same heated lines no hardware change	Medium	UNCHANGED
Avionics/Health Monitoring	Data/Diagn	Receives ECS status for ACMS or BIT reporting	Same data protocols, with additional HPC sensor signals	Medium	MODIFIED to incorporate HPC data streams

System Environment and Constraints

The new Ambient Air Intake ECS operates from sea level to 35,000 ft altitude, with outside temperatures ranging from -60°C to +50°C. Ambient air enters at 0.2-0.3 bar; HPC compresses it to 0.75 bar for cabin supply. Thermodynamic calculations (adiabatic compression) indicate that from a -50°C / 0.25 bar inlet to 0.75 bar, the post-compression temperature rises to 45°C at 80-85% compressor efficiency:

$$T_2 = T_1 \cdot (P_2/P_1)^{((\gamma-1)/\gamma)} \cdot (1/\eta)$$

Table AT.24.: ECS generation report - system environment and constraints

Environmental Factor	Operating Range	Design Constraints	Impact on System	Mitigation Requirements
Altitude	0-35,000 ft cruising	Cabin differential up to ∼8-9 psi	ECS must compress from lower external P	HPC sized for needed compression ratio
Temperature	-60°C to +50°C external	System must handle sub-zero intake & anti-icing measures	Risk of ice ingestion, HPC icing	Intake duct heating or HPC anti-icing logic
Pressure	∼0.2-0.3 bar external at altitude	Requires HPC to achieve ∼0.75 bar cabin supply	Increased electrical load for HPC	Ensure adequate power capacity, HPC surge margin control
Humidity	Varies with external conditions	System must manage condensate once compressed/warmed	Water extraction, dryness level unaffected except HPC can change dew point	Moisture separator, reheater as in original ECS
Electrical Availability	Limited by aircraft bus capacity	Must support HPC startup 15-20 kW transient, 10+ kW steady	Potential upgrade to generator or load-shedding logic	Coordinate HPC drives with overall aircraft power management

Sakeholder Needs Analysis

Table AT.25.: ECS generation report - stakeholder needs analysis

Stakeholder	Primary Needs	Business Drivers	Success Criteria	Priority
Flight Crew	Simple, reliable ECS control and feedback	Safety, workload reduction	Ease of use, stable cabin environment	High
Maintenance Personnel	Effective HPC diagnostic and replaceable modules	Short downtime, cost-effective maintenance	Clear HPC procedures, quick LRU swaps	Medium
Passengers	Comfort. cabin temperature, pressure, humidity	Low flight disruptions, brand reputation	Surveys, negative feed-back minimized	High
Regulatory Entities	Safe HPC integration, verified performance at altitude	Ensure aircraft certification, compliance	All ECS compliance docs approved	High
Aircraft OEM	Scalable HPC-based ECS architect. across multiple platforms	Reduce bleed air usage, align with More Electric Aircraft	Meeting HPC reliability, weight, performance targets	High

Table AT.26.: ECS generation report - key functional requirements

Req. ID	Function Area	Requirement Title	Requirement Statement	Acceptance Criteria	Verific. Method	Reusab. Status
FR-001	Temperat. Control	Maintain Cabin Temperature Range	The ECS shall provide temperature-controlled air (18-24°C) to ensure passenger comfort	Cabin T measured in all zones within ±2°C of setpoint	Test, Simulation, Flight demon.	Directly Reusable
FR-002	Pressuriz. Control	Maintain Cabin Pressure	The ECS shall maintain a differential up to 8-9 psi, achieving ∼0.75 bar cabin pressure at altitude	Cabin pressure remains stable within ±0.2 psi of target	Test, Simulation	Directly Reusable
FR-003	Humidity Control	Remove Excess Moisture	The ECS shall keep humidity within operational limits (separator efficiency >85%)	No condens./ ice in cabin; humidity sensors stable	Environmen tests	Directly Reusable
FR-014	Ambient Intake Compression	Provide HPC for Ambient Air	The ECS shall include an HPC to compress ∼0.2-0.3 bar ambient air to ∼0.75 bar cabin supply with ∼80-85% efficiency	HPC demon. test meets target compression ratio & thermal rise <50°C from -50°C inlet at altitude	Bench HPC test, flight test for real conditions	Newly Introduced
FR-015	Anti-Icing & HPC Protection	Prevent HPC Inlet Icing	The ECS shall ensure HPC inlet remains ice-free by using intake duct heating or alternative deicing strategy	No HPC blade damage or icing in -60°C flight test scenario	Cold-soak test, sensor inspection	Newly Introduced
FR-016	Compres. Surge & Control	Maintain HPC Stability	The ECS shall include a surge margin control algorithm that avoids HPC stall at varying altitude and intake conditions	No HPC stall events in simulated or actual flight envelope tests	HIL simulat., sensor data analysis	Newly Introduced
FR-007	Redund. & Emergency Operation	Update for HPC-based ECS	The ECS shall provide an alternate supply route (either bleed or backup HPC) to maintain safe conditions if primary HPC fails	System transitions to backup within 30s, no cabin pressure drop beyond 0.3 psi below normal	Failure simulat., failover test	Major Modification
FR-008	Health Monit. & Fault Detection	Extend to HPC	The ECS shall continuously monitor HPC performance (speed, torque, T/P) in addition to normal ECS parameters, generating proactive maintenance alerts	HPC health data logged every 1-5 s., ACMS flags anomalies promptly	BIT, ACMS review, HPC sensor checks	Minor Modification

Table AT.27.: ECS generation report - function identification and hierarchy (excerpt)

Func. ID	Function Name	Function Description	Parent Func	Child Functions	Func Level	Change from Original
FN-001	Cabin Environmental Control	Overall top-level ECS function for controlling temperature, pressure, humidity, ventilation	N/A	FN-002, FN-003, FN-004, FN-005, FN-006, FN-007	Level 1 (Root)	un-changed
FN-002	Temperature Control	Regulate supply air temperature to meet cabin setpoints	FN-001	N/A	Level 2	un-changed
FN-003	Pressurization Control	Maintain cabin pressure up to 0.75 bar differential at altitude	FN-001	N/A	Level 2	un-changed
FN-004	Humidity & Moisture Control	Extract moisture, prevent condensation/icing	FN-001	N/A	Level 2	un-changed
FN-006	Redundancy & Emergency Operation	Ability to fail over to alternate HPC or bleed if primary HPC fails; support emergency ram air	FN-001	Failover HPC engine, Emer. ram air	Level 2	Modified from original redundancy approach
FN-007	Health Monitoring & HPC Diagnostics	Monitor HPC speed, T/P, load, plus existing ECS sensors for anomaly detection	FN-001	HPC surge detection, HPC Sensing	Level 2	Modified to include HPC data
FN-008	Ambient Air Intake Compression	Dedicated HPC function to compress 0.2-0.3 bar external air to ∼0.75 bar supply	FN-001	Anti-icing, Surge control subfunctions	Level 2	NEW function

Table AT.28.: ECS generation report - logical element identification (excerpt)

Element ID	Element Name	Element Type	Purpose and Responsibility	Key Attributes	Reusability Assessment
LE-CTRL	ECS Controller	Control & Monitoring	Coordinates modes, collects sensor data, issues commands	Real-time proc., closed-loop controls	Modified for HPC signals
LE-SENS	Sensor Suite	Data Acquisition	Reports T, P, humidity, valve positions, HPC speed	Multi-sensor redundancy	Partially modified with HPC sensors
LE-PRESS	Pressurizat. & Outflow	Pressuriz. Control	Regulates cabin pressure with outflow valve, reading cabin P sensors	Supports 8-9 psi differential	Unchanged
LE-HPC	High-Pressure Compressor Subsystem	New HPC Module	Compress ambient ∼ 0.25 bar to ∼0.75 bar; manage anti-icing, surge margin	Spinner, compr. stages, inlet & outlet sensors, HPC driver	New element (not reusable from bleed ECS)
LE-REDUN	Redundancy Management	Logic & Switching	Manages HPC failover, cross-bleed, or emergency ram air	Maintains cabin environment if HPC fails	Modified ECS function
etc.					

A.13. Evaluation of the Architecture Generation Variants

The evaluation of LLM-generated system architectures reveals heterogeneity in the generation quality of different report sections. While the first two chapters, conceptual foundations and methodological approach, could be created without notable difficulties and exhibited both content consistency and structural coherence, the generation of the third and fourth chapter with quantitative system analyses proved to be considerably more demanding.

Systematic calculation errors characterized the initial generation attempts for architecture evaluation despite correct formula application and proper parameter insertion. Particularly, the thermodynamic calculations of compressor outlet temperatures and the determination of electrical power requirements proved problematic, necessitating iterative refinement of prompt strategies. Only through the implementation of explicit Java calculation instructions could the quantitative reliability of LLM outputs be raised to an acceptable level.

Parallel to the mathematical challenges, the structural organization of architecture generation also proved to require optimization. Experimental variations in the number and segmentation of prompts led to the insight that division into separate generation phases - a dedicated prompt for the first architecture, a second prompt for the second architecture, followed by systematic comparison - leads to qualitatively superior and more consistent results than monolithic generation approaches.

These observations underscore the critical importance of prompt-strategic considerations in LLM-assisted generation of technical content and illustrate that the apparent simplicity of conceptual content is not automatically transferable to quantitative analyses.

Technical-Systematic Evaluation

The first evaluation step is a process-oriented one that examines the quality of the methodology's execution across prompt adherence, mathematical accuracy, and technical understanding.

Evaluation of Prompt Execution Quality

The assessment of the prompt execution quality represents a central category in the evaluation of LLM-based system architecture development. It systematically measures the extent to which the used LLM has followed the explicitly formulated tasks and methodological specifications in the prompt instructions. This evaluation dimension is important for the reproducibility and reliability of LLM-supported engineering processes, as it quantifies the ability for precise implementation of structured development methodologies.

The evaluation was conducted based on four sequential prompt instructions that mapped a complete system architecture development process from Component Reusability Assessment to comparative evaluation of alternative physical architectures. Each prompt contained specific, measurable requirements as well as methodological specifications for documentation structure, mathematical calculations, and transparency marking.

Four central evaluation dimensions were defined for systematic evaluation:

- Instruction adherence captures the accuracy of following specific work instructions. This includes the complete implementation of required deliverables, adherence to technical specifications, and implementation of special methodological requirements such as thermodynamic calculations or iterative design processes.

- Format consistency evaluates adherence to prescribed documentation structures and standards. The consistent application of table formats, systematic outline structure, and consistent use of technical documentation standards across all development phases are analyzed.

- Completeness examines the completeness of deliverable creation. It evaluates whether all explicitly required components were created, all task aspects were processed, and all specified analysis steps were performed.

- Color coding evaluates the consistent application of the prescribed transparency system for marking information origin. The implemented system distinguishes between reused (green), modified (orange), new (blue), and assumed (red) elements and explicitly requires that "No component or specification should appear without color coding."

Table AT.29.: Quantitative assessment of prompt execution quality dimensions

Assessment Dimension	**Score**	**Key Findings**
Instruction Adherence	5/5	Complete adherence to all prompt requirements across four sequential development phases. Precise implementation of specific requirements such as mandatory thermodynamic calculations, independent problem detection, and fundamentally different architecture approaches.
Format Consistency	5/5	Consistently uniform HTML structuring with uniform performance tables, systematic organization, and standardized Traceability Matrices across all reports.
Completeness	5/5	Complete creation of all explicitly required deliverables: Component Reusability Matrix, Complete Requirements Specification, Functional/Logical Design, two Physical Architectures with Comparison Analysis. No content gaps identified.
Color Coding	4/5	Initially excellent application of the transparency system in early development phases (Component Reusability, Requirements Specification) with systematic four-color marking. Quality declined during architecture development phases, where focus shifted to thermodynamic calculations - green coding was maintained, but blue/orange/red markings became inconsistent.
Overall Rating	**4.75/5**	**Excellent Prompt Execution Quality with minor inconsistencies in late-phase transparency marking**

The overall rating (4.75/5), as can be seen in Table AT.29, demonstrates a high prompt execution quality. Particularly noteworthy is the precise interpretation and implementation of complex, multi-level prompt structures with multiple mandatory requirements.

The largely consistent application of the transparency system through color coding, particularly in early development phases, demonstrates the potential for systematic traceability of design decisions and information sources; however, some inconsistencies emerged during the calculation phase. The complete creation of all required deliverables without identified content gaps underscores the general suitability of the LLM approach for structured engineering documentation tasks, while highlighting the need for enhanced prompt design to maintain transparency standards throughout all development phases.

The results show that advanced LLMs can achieve high reliability in task implementation with precisely formulated prompt instructions. This is particularly important for standardization and quality assurance in LLM-supported development processes, as it enhances the predictability and reproducibility of generated results, thereby establishing the methodological foundation for productive use in system architecture development.

Evaluation of Mathematical Accuracy and Feasibility

The assessment of mathematical accuracy and feasibility analyzes the correctness and plausibility of technical content in LLM-generated system architectures. This dimension captures the ability to apply physical laws, perform mathematical calculations, and apply engineering principles accurately without external correction or validation.

The evaluation was conducted based on thermodynamic calculations, performance analyses, and system designs in both developed ECS architectures. Special attention was paid to the independent application of the adiabatic compression formula, pressure budget analyses, and power calculations.

Four different points were examined more closely in this category:

- Formula Application analyzes the correct application of formulas.
- Mathematical Execution captures the computational correctness of performed calculations. Numerical accuracy, unit consistency, and correct implementation of mathematical operations are evaluated.
- Calculation Traceability evaluates the comprehensibility and transparency of calculation paths. The documentation quality of mathematical steps, assumptions, and boundary conditions are analyzed.
- Engineering Consistency captures the consistency of technical specifications across all system levels. The coherence between individual calculations and overall system design is evaluated.

The evaluation of Mathematical Accuracy requires a differentiated consideration of prompt engineering challenges in complex mathematical calculations. While the final results demonstrate high accuracy, the development of suitable prompt structures for accurate calculation execution was an iterative process that yielded significant methodological insights.

Early prompt versions with explicit formula specification led to systematic calculation errors despite correct parameter substitution. A characteristic example was shown in the

Table AT.30.: Quantitative assessment of mathematical accuracy and feasibility

Assessment Dimension	Score	Key Findings
Formula Application	5/5	Correct application of adiabatic compression formula and electrical power calculation formula. Systematic use of thermodynamic fundamentals for all compression stages.
Mathematical Execution	5/5	Computationally correct implementation of all thermodynamic calculations. Consistent unit usage (bar, °C, kW) across all calculations.
Calculation Traceability	5/5	Completely traceable calculation documentation with explicit formulas, parameters, and intermediate steps. Systematic verification of all calculations with documented assumptions and boundary conditions.
Engineering Consistency	5/5	Coherent technical specifications across all system levels. Pressure budget analysis is consistent from the HPC output to the Mixing Manifold. Mass flows, energy balances, and component designs are system-wide coordinated.
Overall Rating	**8/5**	**Excellent mathematical accuracy and engineering plausibility**

calculation of HPC Stage 1 outlet temperature, where the LLM arrived at an incorrect result despite completely correct parameter substitution:

$$T_2 = T_1 \cdot \left(\frac{p_2}{p_1}\right)^{\left(\frac{\kappa-1}{\kappa}\right)} \cdot \frac{1}{\eta} \tag{AE.1}$$

$$= 223\text{K} \cdot \left(\frac{0.50}{0.25}\right)^{0.286} \cdot \frac{1}{0.85} \tag{AE.2}$$

$$= 320\text{K} \tag{AE.3}$$

However, the LLM incorrectly calculated 263K (-10°C) instead of the correct 320K (47°C). Remarkably, the subsequent Kelvin-Celsius conversion (263K - 273K = -10°C) was performed correctly, indicating that the LLM works more reliably with simple arithmetic operations than with more complex exponential function calculations.

Calculation accuracy was achieved only through the explicit integration of JavaScript calculation scripts into the prompt structure. This methodological adaptation enabled consistent thermodynamic calculations and formed the basis for the subsequent evaluation of the architectures.

The high rating (5/5) demonstrates the mathematical accuracy and engineering plausibility of LLM-generated calculations following methodological optimization of the prompt design. However, the correct application of complex thermodynamic formulas was only achieved after integration of structured calculation tools, which has important implications for LLM-based quantitative engineering applications.

The systematic pressure budget analysis, accompanied by independent problem recognition and correction, demonstrates genuine engineering problem-solving competence. The

power calculations utilize realistic efficiency parameters and account for multi-stage energy conversion chains.

The results demonstrate that advanced LLMs can correctly handle complex mathematical calculations and physical relationships when suitable methodological structures (Analysis Tools) are implemented. This is an essential insight for productive use in quantitative engineering applications, underscoring the importance of prompt engineering for mathematical precision.

Evaluation of Technical Understanding

The assessment of the technical understanding analyzes the engineering competence of LLM-generated system architectures beyond structural prompt compliance. While aspects such as documentation format and process adherence can be controlled through explicit instructions, technical competence manifests in the ability to recognize independent problems, design systematic improvements, and make well-founded decisions without external guidance.

Five dimensions were defined for systematic evaluation, capturing central aspects of engineering problem-solving competence, and are evaluated based on the developed ECS architectures.

- Problem identification capability captures the ability for independent recognition of technical problems, inconsistencies, or specification deviations without external hints. This competence is shown in the automatic detection of design deficits and the independent initiation of corrective measures.
  - The LLM independently recognized the critical pressure drop in the first architecture: *"Observing the shortfall in final manifold pressure (0.76 vs. required $\geq$0.80 bar) triggered a design revision: HPC ratio $\sim$4.0 was insufficient once we accounted for a $\sim$0.20-0.24* bar *total drop downstream."*
- Design iteration quality evaluates the ability for systematic, targeted improvement of system designs based on recognized deficits. The structuredness of the problem-solution cycle and the comprehensibility of improvement logic are analyzed.
  1. *Problem Analysis: HPC exit at 1.0* bar *left only $\sim$0.76* bar *at the manifold after subsystem losses.*
  2. *Solution Evaluation: Systematic evaluation of three options (HPC pressure increase, additional compression stage, loss reduction)*
  3. *Modification Decision: Implement Option A, single-stage HPC at $\sim$1.1* bar *discharge. Accept the slightly higher HPC temperature and power load.*
  4. *Implementation Impact: HPC exit $\sim$140°C instead of $\sim$130°C. Large Final Cooler must handle an extra 10°C. Slightly higher HPC power needed.*
  5. *Verification: New HPC ratio = 4.4 meets final 0.8 bar. Pressure and temperature calculations were repeated to confirm viability.*
- Trade-off analysis and justification encompasses the ability for systematic evaluation and justification of design decisions through explicit weighing of competing requirements and constraints. This includes quantitative comparison analyses and transparently documented decision logic.

  - Systematic architecture comparisons with quantitative metrics: *"For an all-electric ECS, Architecture 1 has lower total HPC power but yields higher HPC outlet temperature. Architecture 2 suits scenarios prioritizing lower HPC stage temperatures (potentially better reliability or improved cooler life)."*

- Design Rationale Documentation captures the quality of design decision documentation with explicit justification logic and direct linkage to system requirements and physical constraints.
  - Transparent documentation of all design decisions such as: *"For an all-electric ECS, Architecture 1 has lower total HPC power but yields higher HPC outlet temperature. Architecture 2 suits scenarios prioritizing lower HPC stage temperatures (potentially better reliability or improved cooler life)."*
- System-Level Thinking evaluates the ability for holistic system consideration with understanding of interdependencies and cascade effects between components and subsystems.
  - System-Level Thinking competence manifests in comprehensive Component-Reusability Analysis with systematic evaluation of interdependencies: *"Key Findings: Many core elements (e.g., distribution ducts, certain valves) can be reused with modifications, but the compression stage requires new or heavily modified components (due to lower inlet pressure and different temperature profile)."* Systematic Interface Transition Analysis and Cross-System Dependencies across all subsystems.

The quantitative results of this assessment are summarized in Table AT.31.

The rating (4.9/5) in all dimensions demonstrates the technical competence of the used LLM. Particularly noteworthy is the ability to recognize independent problems and develop systematic solutions without external hints, which indicates genuine engineering problem-solving competencies.

The comprehensive Component Reusability Matrix and systematic Interface Transition Analysis demonstrate a holistic understanding of the system that extends beyond isolated component optimization. The LLM systematically analyzed primary, secondary, and cascade effects of HPC integration on all subsystems.

The results show that LLM-based systems can develop genuine engineering problem-solving competencies with appropriate methodological guidance that qualitatively differ from mechanical rule application. This has far-reaching implications for the integration of AI systems in complex engineering processes and the development of collaborative human-AI approaches in systems engineering.

Elegant Design Evaluation after Watson and Griffin

The evaluation of LLM-generated ECS architectures according to Watson and Griffin's Elegant Design Framework complements the process-oriented evaluation with the important dimension of solution quality. While the previous categories primarily evaluated the working method of the LLM (Prompt Execution, Mathematical Accuracy), Watson's Framework analyzes the engineering quality of the generated system solutions themselves.

Watson et al. [179] define elegant engineering solutions through four central attributes: Effectiveness (fulfillment of functional requirements), Robustness (performance under varying conditions), Efficiency (resource optimization), and Minimization of unintended consequences (avoidance of negative system interactions). These evaluation dimensions enable a systematic analysis of LLM architectures compared to established reference systems.

Table AT.31.: Quantitative assessment of technical understanding dimensions

Assessment Dimension	**Score**	**Key Findings**
Problem Identification	5/5	Excellent independent problem recognition: Independent identification of pressure drop in Architecture 1 (0.76 vs. 0.80 bar requirement) with documented analysis. Systematic root-cause analysis without external prompts.
Design Iteration Quality	5/5	Systematic design improvement with completely documented iteration cycles. Structured Problem-Identification → Solution-Evaluation → Implementation → Verification for both architectures. HPC adjustment from 1.0 to 1.1 bar with thermodynamic recalculation.
Trade-off Analysis	5/5	Comprehensive quantitative comparison analysis between both architectures: Single-stage (∼400 kW) vs. Multi-stage (∼570-600 kW) HPC with explicit evaluation of power requirements, system complexity, thermal aspects, and Implementation Risk Assessment.
Design Rationale	5/5	Transparent documentation of all design decisions with explicit justification logic. Complete Decision Traceability Matrix and Technical Problem Resolution Log. Thermodynamic calculations are systematically documented and verified.
System-Level Thinking	5/5	Outstanding holistic system consideration with Component Reusability Matrix across all subsystems. Systematic Interface Transition Analysis and Cross-System Dependencies. Understanding of cascade effects between HPC integration and downstream systems.
Overall Rating	**5/5**	**Excellent technical competence with systematic engineering thinking processes**

Effectiveness (Function Fulfillment)
Both LLM architectures completely fulfill the specified system requirements: 0.8 bar manifold pressure, <25°C air temperature, and 1.8 kg/s mass flow. The systematic pressure budget analysis ensures that sufficient safety margins are maintained across all subsystems. Compared to the B787 reference, both concepts demonstrate functional equivalence, but without operational validation proof from the production aircraft.

Robustness (System Stability)
Architecture 1 offers inherent robustness through its simplicity, with fewer potential failure modes; however, it is vulnerable to temperature spikes with single-stage compression. Architecture 2 improves thermal stability through intercooling but increases system complexity with additional failure paths. The B787 eECS sets the standard with dual-redundant systems and regenerative cooling, providing exceptional operational stability.

Table AT.32.: Quantitative assessment according to the elegant design framework

Criterion	**Architecture 1 (Single-stage HPC)**	**Architecture 2 (Multi-stage HPC)**	**B787 eECS (Reference)**
Effectiveness	**4/5** Fulfills all specifications for pressure, temperature, and mass flow. Direct goal achievement with simple system architecture.	**4/5** Identical performance fulfillment with improved temperature control. Same functional coverage.	**5/5** Complete fulfillment of all cabin environment requirements. Proven in operational use with >10 years of operating experience.
Robustness	**3/5** Simple architecture with fewer failure modes, but vulnerable to HPC temperature spikes under extreme conditions. Single-point-of-failure in HPC.	**4/5** Improved thermal stability through intercooling. More complex architecture with more failure modes, but better degradation with partial failures.	**5/5** Dual-redundant systems with proven operational stability. Comprehensive fail-safety through regenerative cooling and turbine-assisted expansion.
Efficiency	**4/5** Lowest power requirement (∼400 kW) and lowest system complexity. Acceptable power density for electric ECS systems.	**2/5** High power requirement (∼570-600 kW) due to double compression stages. Intercooler additional weight reduces overall efficiency.	**5/5** Optimized thermodynamic cycles with regenerative heat recovery. Elimination of engine bleed improves Engine efficiency by ∼3 %.
Minimization of Unintended Consequences	**3/5** High HPC outlet temperature (140°C) heavily loads Final Cooler. Integration into existing aircraft architectures challenging due to high single-component power.	**3/5** Intercooler requires additional packaging solutions and Ram-Air capacity. More complex control logic with potential control instabilities between HPC stages.	**5/5** Fully integrated system solution without critical system interactions. Reduced engine bleed requirements improve Engine lifespan.
Overall Rating	**3.5/5**	**3.25/5**	**5/5**

Efficiency (Resource Optimization)
The efficiency comparison reveals significant differences: Architecture 1 minimizes power requirements ($\sim$ 400 kW) and system complexity, while Architecture 2 exhibits significantly higher energy demand ($\sim$ 570-600 kW) due to the presence of double compression stages. The B787 eECS demonstrates superior overall efficiency through regenerative heat recovery and elimination of engine bleed, improving Engine efficiency by $\sim$3 %.

Minimization of Unintended Consequences
Both LLM architectures show typical challenges of new system concepts: Architecture 1 loads the Final Cooler through high HPC outlet temperatures, while Architecture 2 creates additional packaging requirements for the intercooler. The B787 eECS benefits from complete system integration without critical negative interactions and even reduces engine loading.

The evaluation shows that both LLM-based architectures represent functionally adequate but not optimal engineering solutions. With overall ratings of 3.5/5 and 3.25/5, respectively, they achieve solid engineering quality but remain significantly behind the optimized B787 reference (5/5).

Architecture 1 offers the most balanced solution with acceptable efficiency and the lowest complexity, but suffers from thermal challenges. Architecture 2 significantly improves temperature control, but it comes at a high price in terms of energy demand and system complexity.

The evaluation demonstrates an essential insight for LLM-based engineering applications: Excellent process quality (5/5 in Technical Understanding) does not automatically lead to optimal engineering solutions. LLMs can work systematically and perform correct calculations, but do not yet achieve the optimization depth and system integration of experienced engineering teams.

This underscores the importance of hybrid approaches where LLMs are used for structured development work and concept generation, while critical optimization and system integration still require intensive human expertise.

A.14. Fuel Cell Systems in Aviation

This chapter provides technical background on fuel cell systems and thermal management in aviation applications. While not central to the methodological focus of this dissertation, this material offers essential context for understanding the technical evaluation case presented in Chapter 6.

The current system architecture of civil aircraft relies on the use of electrical, hydraulic, and pneumatic power to supply various aircraft systems. This fundamental architecture has remained essentially unchanged over the past decades. However, there is currently a clear trend toward replacing hydraulically and pneumatically supplied aircraft systems with more energy-efficient electrical systems, which is subsumed under the term *More Electric Aircraft* (MEA), leading to an increase in electrical power requirements on board. This shift toward electrified system architectures is reflected in modern aircraft designs, including the Airbus 350 and 380, as well as Boeing's 787 [1], [195, p.1], [199, p.2]. A currently often discussed solution for providing the required electrical power is the use of fuel cell systems (FuCS). FuCS converts the electrochemical energy of hydrogen[1] and oxygen into electrical energy, water, and heat, operating without the emission of pollutants or noise [1], [199, p.2].

Research on integrating fuel cell systems into aircraft architecture has been ongoing for several years. Worldwide, there are a total of two aircraft that can be powered entirely by a fuel cell system throughout the flight. These are the *Antares* from the *German Aerospace Center*, with a fuel cell power output of up to 45 kW, and a six-seater Piper M350 converted to fuel cells by *ZeroAvia*, with a power output of 250 kW. Both aircraft are propeller-driven. Since significantly higher power classes in the megawatt range are required to power a commercial aircraft, numerous investigations are currently underway regarding the system-technical integration of fuel cell systems [199, p.2]. Airbus, together with Michelin and DLR, successfully developed a fuel cell system in 2008 and integrated it as an emergency power system in an A320 [223, p.436]. The system was successfully tested during cruise flight in various acceleration and pitch maneuvers. In 2012, Boeing, in collaboration with Ishikawajima-Harima Heavy Industries (IHI), tested a regenerative fuel cell system integrated into a Boeing 737. The system can produce hydrogen through electrolysis processes during flight phases when the full electrical power is not needed. The electrical power output of the fuel cell system was not specified [223, pp.435-437].

There are multiple approaches to integrating fuel cell systems into aircraft architecture, whether for the propulsion system, as an emergency power source in the event of a power failure (RAT replacement), or as a permanent power supply for aircraft systems. The range of applications is broad, leaving room for creative solutions in aircraft development. However, to gain a deeper understanding of the potential integration, it is first essential to examine fuel cell system technology more closely.

Classification of Fuel Cells

A fuel cell (abbreviated as FuC) is a galvanic cell that generates electricity through an electrochemical reaction [224, p.30] and can be differentiated into five different fuel cell types depending on the electrolyte used and temperature range [197, p.13f], [225, p.9]. The naming of the different fuel cell types is based on the electrolyte used [225, p.9]. The respective fuel cell types are listed in Table AT.33, along with their most essential characteristics.

[1] If the hydrogen is generated using renewable energy, this technology can be classified as environmentally friendly since no pollutants are emitted

Table AT.33.: Overview of fuel cell types and their characteristics [203, p.8], [205, p.7]

Fuel Cell Type	**Alkaline Fuel Cell**	**PEM Fuel Cell**	**Phosphoric Acid Fuel Cell**	**Molten Carbonate Fuel Cell**	**Solid Oxide Fuel Cell**
Classification	Low temp.	Low temp.	Medium temp.	High temp.	High temp.
Electrolyte	Potassium hydroxide	Proton-conducting membrane	Phosphoric acid	Molten alkali carbonate	Zirconium dioxide
Charge carrier	OH^-	H^+	H^+	CO_3^{2-}	O^{2-}
Operating temp. in °C	60 - 90	80	200	650	600 - 1000
Catalyst	Platinum	Platinum	Platinum	Nickel	Nickel-zirconium oxide
Fuel gas	Pure H_2	Pure H_2	H_2 reformate	H_2, CH_4	H_2, CH_4, CO
Oxidant	Pure O_2	O_2, air	O_2, air	O_2, air	O_2, air
El. efficiency in %	60 - 70	50 - 68	45 - 55	65	60 - 65

For use in mobile applications such as commercial aircraft, **P**olymer **E**lectrolyte **M**embrane (PEM) fuel cells have become established. This is due to their simplicity in design, robustness against vibrations, and high technological maturity [198, p.6764]. Furthermore, they exhibit excellent start/stop behavior, are easy to control, and have short response times to load changes of less than 1s. PEM fuel cells demonstrate good cold-start behavior at temperatures down to -20°C, and despite high forces occurring due to varying ambient pressures, operational vibrations, and accelerations in the aircraft, the operation of the fuel cell is secure due to the use of a solid electrolyte. Moreover, PEM fuel cells can be operated in any orientation, making them particularly attractive for aircraft applications. They operate in a temperature range between 60 and 80°C, demonstrate an electrical efficiency of 50-68% per cell, and additionally possess a comparatively high power density of up to 1000 W/kg [197, p.18], [199, p.9], [201, p.77], [202, p.2], [203, p.282, 390]. When referring to a fuel cell (FuC) in the following text, it always refers to a PEM fuel cell.

Operating Principle of a PEM Fuel Cell

As shown in Table AT.33, the fuel cell uses the reaction gases hydrogen H_2 and oxygen O_2 to generate electrical energy in the form of direct current from the chemical energy of both reaction gases [200, p.28], [226, p.5]. The structure and operating principle are explained below using Figure AF.5.

The fuel cell is composed of two gas-diffusion electrodes, the anode and cathode, which are separated by a proton-conducting membrane that is between 50 and 150 μm thick and made of perfluorinated sulfonated polymers [201, p.88f]. The membrane is permeable to protons and other cations but not to electrons [200, p.29]. The reaction gases, hydrogen

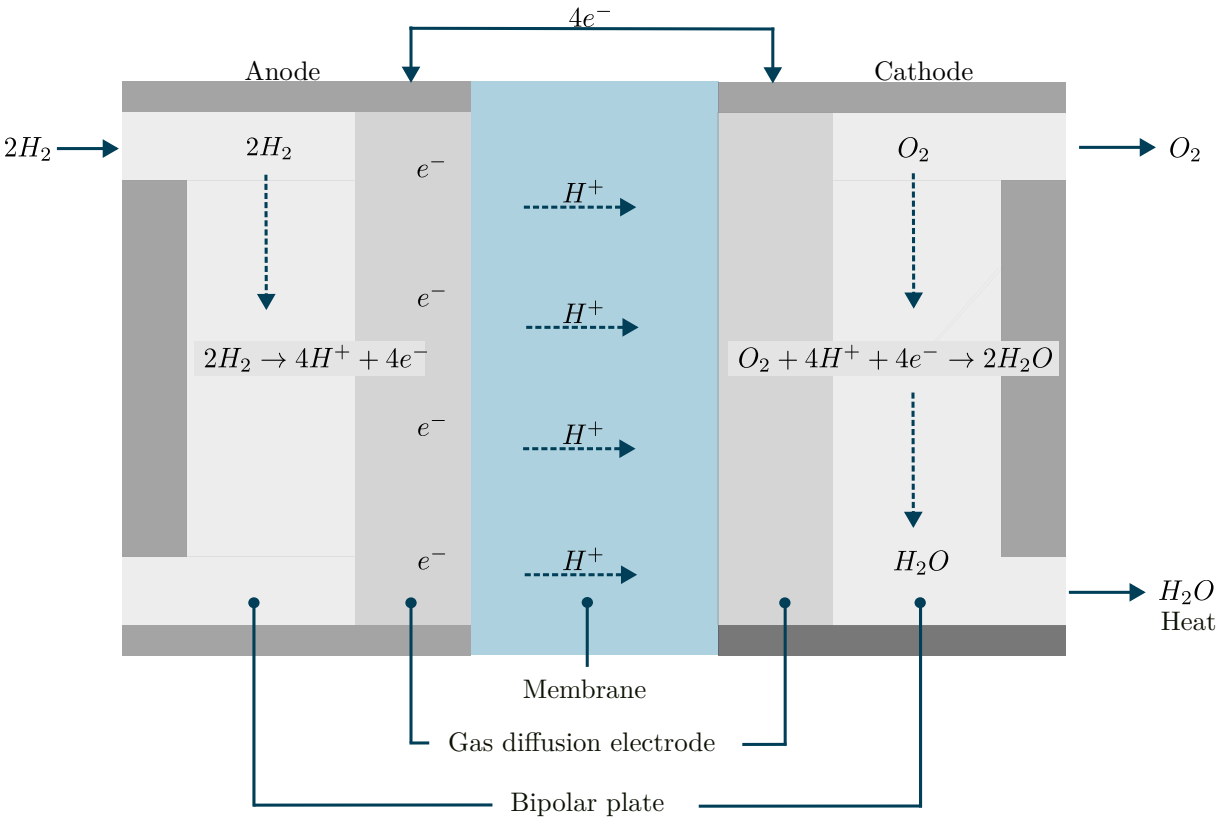

Figure AF.5.: Schematic structure of a PEM fuel cell, adapted from [200, p.29], [203, p.119]

and oxygen, are supplied over-stoichiometrically[2] through the gas distribution structure of the two bipolar plates and directed to the corresponding gas diffusion electrodes, which are typically made of porous carbon and must exhibit good electrical conductivity, gas tightness, corrosion stability, as well as pressure and bending strength [201, p.6 and p.94], [205, p.7], [200, p.29]. To enhance the electrochemical reactions taking place at the interface between electrode and membrane, both electrodes are coated with an additional catalyst layer (typically platinum particles) [200, p.29]. On the anode side, pure hydrogen is supplied through the bipolar plate, resulting in the following hydrogen oxidation reaction at the interface between anode and membrane [226, p.5]:

$$2H_2 \rightarrow 4H^+ + 4e^- \tag{AE.4}$$

This reaction produces both positively charged hydrogen atoms (H^+), which diffuse directly to the cathode through the proton-conducting membrane, and negatively charged electrons (e^-), which are conducted to the cathode via an external electrical circuit [201, p.6], [205, p.8]. This electron flow, which induces an electrical direct current, is caused by reaction-related potential differences between anode and cathode [199, p.11], [227, p.5], [197, p.14]. The electrical direct current I generated by this electrochemical reaction is proportional to the Faraday constant F[3] and the electrons flowing in the external circuit

[2]The two reaction gases are supplied to the fuel cell over-stoichiometrically to ensure adequate supply. This means that a larger mass flow of hydrogen and oxygen is supplied than is actually necessary for the electrochemical reaction [199, p.11]

[3]The Faraday constant F describes the amount of electrical charge contained in one mole of electrons and has the constant value of $96,485\ \mathrm{Cmol}^{-1}$ [226, p.23]

$\dot{n}_{El}$, where:

$$I = F \cdot \dot{n}_{El} \tag{AE.5}$$

For each hydrogen molecule (H_2), a total of two electrons are transported to the cathode [197, p.14f]. At the cathode, the following oxygen reduction takes place [226, p.5]:

$$O_2 + 4H^+ + 4e^- \rightarrow 2H_2O \tag{AE.6}$$

The oxygen supplied through the bipolar plate reacts there with the positively charged hydrogen atoms and the released electrons to form water (H_2O) [200, p.29f]. As shown in Figure AF.5, the produced water is removed along with the oxygen-reduced air through the gas distribution structure of the bipolar plates [200, p.29f], [226, p.6].

Voltage-Current Density Diagram

The electrical operating behavior can be represented in a voltage-current density diagram for stationary operating points, see Figure AF.6 [199, p.11]. For better representation independent of cell size and power class of the considered fuel cell, this diagram uses the current density i, which is formed by the quotient between the electrical current I and the electrode area A_Z and is defined as follows [199, p.11], [200, p.30], [197, p.15], [203, p.80]:

$$i = \frac{I}{A_Z} \tag{AE.7}$$

As shown in Figure AF.6, a maximum theoretical cell voltage of U^0 can be generated per fuel cell. This cell voltage results from the energy released during the electrochemical

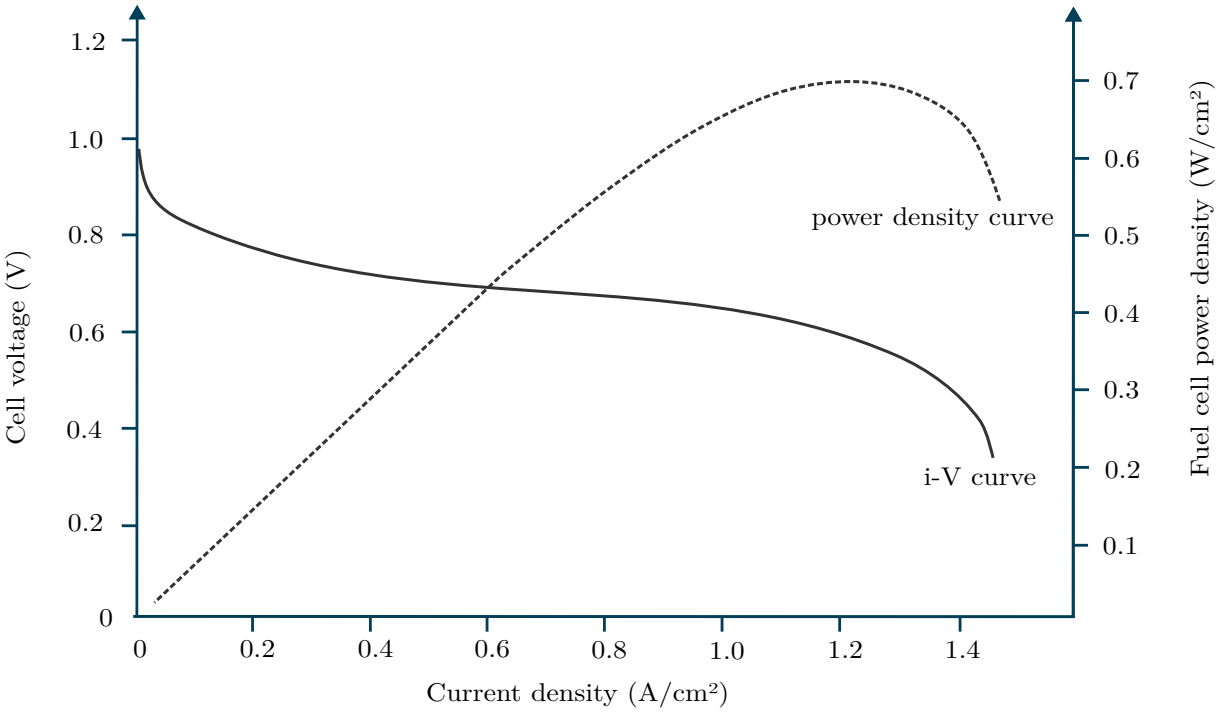

Figure AF.6.: Schematic progression of a voltage-current density characteristic curve, adapted from [203, p.19]

reaction and is calculated as follows:

$$U^0 = -\frac{\Delta G^0}{z \cdot F} \tag{AE.8}$$

Here, ΔG^0 is the molar free reaction enthalpy of the gaseous water formed, which is -228.57 kJ/mol at ambient conditions, z is the number of electrons released per oxidation, and F is the Faraday constant. Since each hydrogen atom releases two electrons during oxidation at the anode, $z = 2$ [199, p.11], [200, p.29f]. Based on these three parameters, a theoretically achievable reversible cell voltage of $U^0 = 1.18$ V results, which would ideally remain constant over the current density, see Figure AF.6 [199, p.11].

In practice, however, the electrical output voltage decreases continuously as the current density increases. As shown in Figure AF.6, the open-circuit voltage measured in the fuel cell is already significantly lower than the theoretically achievable reversible cell voltage of 1.18 V. It typically measures around 1 V and is attributable to so-called mixed potential losses, which are caused by the diffusion of hydrogen and oxygen molecules as well as occasional electrons through the membrane [79, p.21]. Furthermore, additional losses occur depending on the current density, which are caused by four essential operating principles, as can be seen in Figure AF.7 and can be divided into three regions:

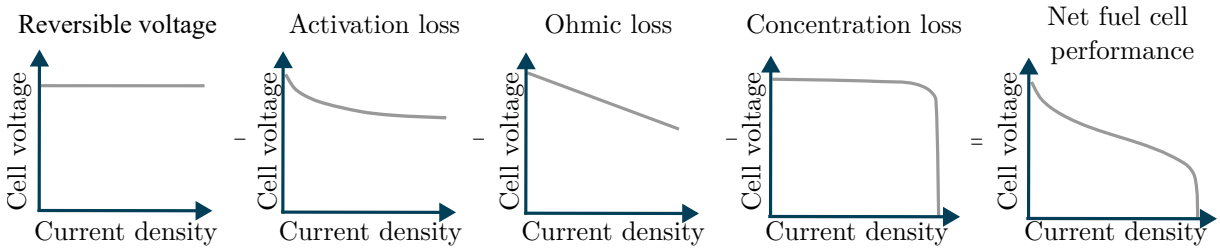

Figure AF.7.: Schematic progression of a i-A curve with the various loss mechanisms occurring during operation, adapted from [203, p.205]

1. The Activation Region:
In this region, the cell voltage drops due to irreversible reaction kinetic losses at the electrodes [201, p.28], [205, p.14]. This is caused by the fact that a specific thermal activation energy (activation loss) is necessary for breaking and forming chemical bonds, which has not yet been reached at this point. According to the Butler-Volmer equation, activation losses increase nearly exponentially with current density [228, p.14]. Additional losses may also occur in this region due to side reactions caused by impurities in the reaction gases [205, p.14]. To minimize these losses, the electrodes are coated with catalysts to reduce the activation energy required.

2. The Working Region:
With increasing current density, ohmic losses dominate in the second region, showing an approximately linear relationship with current density, and are caused by resistance forces against the charge carriers in the fuel cell. Here, particularly the positively charged hydrogen atoms experience a resistance force $R_{Electrolyte}$ during their transport through the membrane. The degree of membrane hydration plays a major role in lowering internal resistance. When this is around 30%, good proton conductivity is ensured and ohmic losses are minimized [201, p.79], [199, p.12]. The membrane's moisture content depends on two operating principles: On the cathode side, water diffuses through the membrane to the anode due to concentration differences. This water either comes from the oxygen reduction at the cathode or is supplied to the cathode air before entering the fuel cell. In the opposite direction, positively charged hydrogen ions migrate from the anode across

the membrane toward the cathode, leading to membrane dehydration [229, p.76], [228, p.15]. Since membrane humidification depends on the cathode air's moisture content, the air humidity must be above 80% to prevent excessive drying. However, too high air humidity is also not desirable, as liquid water can accumulate in the porous electrodes and block the mass transport of reaction gases [228, p.15], [229, p.80]. Another critical parameter is the operating temperature, as the following applies: The higher the fuel cell temperature, the more water the cathode air can absorb, resulting in a lower moisture content. Since the relative moisture content in the membrane is achieved more quickly with higher operating pressure, FuCS delivering an electrical power of more than 5 kW and an operating temperature above 80°C are operated at pressures around 2 bar [229].

3. The Limiting Current Region:
In the region of high current densities, a rapid voltage drop is observed, caused by a concentration decrease of the reaction gases. Here, the losses are caused by too slow transport of reaction gases in the porous electrodes of the fuel cell, where the transport is limited by the diffusion speed in the electrodes [225, p.14]. Consequently, the partial pressure decreases, which simultaneously leads to a reduction in voltage, see Figure AF.7 [201, p.28], [199, p.12], [204, p.111], [228, p.14].

In summary, the progression of the characteristic curve depends on the working pressure, membrane humidification, operating temperature, and the hydrogen-air ratio [227, p.6].

Structure of a Typical PEM Fuel Cell System

Supplying the onboard systems with sufficient electrical power requires voltages higher than the 0.6-0.7 V typically produced by a single fuel cell [203, p.347]. Therefore, for this application area, individual cells are connected in series, causing the individual voltages to add up and the total power to increase while the current density remains constant [226, p.7], [227, p.5], [205, p.8]. Such an assembly of many individual fuel cells is referred to as a fuel cell stack.

Fuel Cell Stack

In total, between 250 and 300 cells are connected in this way to generate technically usable voltages of more than 200 V. Such a fuel cell assembly is referred to as a fuel cell stack [227, p.5].In this configuration, the double-sided, milled, and electrically conductive bipolar plates, typically made of graphite or metal, are alternately connected in series with an electrode-membrane assembly, also known as a MEA (Membrane Electrode Assembly). The flow field etched onto the bipolar plates ensures a uniformly laminar supply of both reaction gases, hydrogen and oxygen. Simultaneously, the product water generated through the electrochemical reaction is removed from the cell via the flow channels [202, p.9], [201, p.94]. Through the double-sided electrical contact, the bipolar plate between two MEAs serves as an anode for one cell and as a cathode for the other [228, p.15].

To extract the generated electrical power from the stack, current collector plates are mounted on the two outermost bipolar plates. These ensure that the generated current is conducted into the external circuit and is available for the electrical consumers aboard the aircraft. The stack is held together using end plates that are clamped together [226, p.7]. Such a PEM stack achieves an electrical efficiency between 50 and 68% and operates at an operating temperature of nearly 80°C [201, p.68f]. For power generation, additional subsystems beyond the actual fuel cell stack are necessary, collectively referred to as the fuel cell system (FuCS), which increases system complexity, weight, and parasitic power

consumption [79, p.23]. The following subsystems are distinguished: hydrogen and oxygen processing, the water module, the cooling system, and the power electronics.

Hydrogen Supply

PEM fuel cells require gaseous hydrogen as fuel. The gravimetric energy density of hydrogen, defined as energy content[4] per mass, is 33.3 kWh/kg and shows approximately three times higher energy density compared to conventional fuels like gasoline (12.1 kWh/kg) or diesel (11.8 kWh/kg) [201, p.230], [228, p.18]. Since hydrogen is the element with the lowest atomic mass, its volumetric energy density[5] is very low. Consequently, hydrogen must be carried in large quantities aboard to ensure a continuous supply to the fuel cell during flight. Furthermore, the gravimetric energy density drops rapidly when the weight of the storage system is included in the calculation. Three principal forms of hydrogen storage are distinguished, as outlined in Table AT.34 and briefly explained below.

Table AT.34.: Comparison of selected hydrogen storage forms [228, p.18], [205, p.7]

Status of Hydrogen	**Efficiency**	**Tank Energy Density**	
	kg H_2/kg Tank	Gravimetric kWh/kg	Volumetric kWh/L
Gaseous H_2 at 300 bar	3.1 %	1.2	0.55
Gaseous H_2 at 700 bar	4.8 %	1.9	1.3
Metal hydride	0.65 %	0.26	1.12
Liquid H_2 at 15 K	14.2 %	5.57	1.68

1. Compressed Gaseous Hydrogen:
To carry gaseous hydrogen in sufficient quantities aboard, special pressure vessels have been developed in which hydrogen is stored under high pressure [199, p.14]. Typical storage pressures range from 200 to 700 bar, with the compression of hydrogen requiring a significant energy input. Approximately 10% of the hydrogen energy content must be expended to compress the gas to 300 bar [203, p.359]. It should be noted that hydrogen storage under high pressure can pose significant safety concerns [199, p.14]. In the aviation industry, the use of pressure vessels is only suitable for systems like the Ram-Air Turbine, which perform emergency functions aboard [199, p.14]. These systems are typically inactive and only require hydrogen in emergencies.

2. Metal Hydride Storage:
Metal hydrides consist of metal alloys that are ground into a fine powder and housed in a solid, easily manageable metal container. At specific pressure and temperature levels, large quantities of hydrogen atoms can be absorbed by the metal powder, releasing heat in the process. By heating the metal hydrides, the hydrogen bound in the powder is rereleased and becomes available for the fuel cell's hydrogen supply.

As Table AT.34 shows, this process can achieve very high volumetric storage densities. However, it should be considered that the metal alloys have a high inherent weight, resulting in comparatively low gravimetric storage energy density. Due to this low storage energy density, this storage system is currently not suitable for aviation applications. In the future, this storage form could represent an interesting alternative through novel

[4] 1 kWh = 3.6 MJ
[5] Measure of material energy per volume with the unit kWh/m^3

organic materials and associated higher energy density [203, p.359f], [199, p.13], [228, p.18].

3. Cryogenic Hydrogen Tank:
To store hydrogen at both high gravimetric and volumetric energy densities, it is cooled to temperatures below 22 K (-251°C). The hydrogen condenses, enabling storage in special tanks at low pressures. A major advantage of this storage form is that it shows the highest storage efficiency of 14.2 % relative to tank mass compared to all other options mentioned. The gravimetric energy density is approximately five times higher than that of gaseous hydrogen storage, reaching nearly 5.6 kWh/kg, as shown in Table AT.34. A disadvantage is the high energy content of approximately 30% that must be expended for condensation, as well as the construction of the cryogenic hydrogen tank. Given its favorable energy density in both mass and volume, cryogenic hydrogen storage is particularly well-suited for use in the aircraft sector.

The hydrogen quantity $n_{H_2,FuC}$ required for the electrochemical reaction depends on the electrical current I to be generated by the fuel cell. The current represents the quantity of electric charge Q passing through a conductor per unit time t, where:

$$I = \frac{Q}{t} \tag{AE.9}$$

According to Faraday's first law, which describes the relationship between electrical charge Q and material requirement, the required hydrogen quantity $n_{H_2,FuC}$ is directly proportional to Q, where:

$$n_{H_2,FuC} = \frac{Q}{z \cdot F} \tag{AE.10}$$

Using equation AE.9 solved for Q, a required hydrogen molar flow rate per fuel cell results in:

$$\dot{n}_{H_2,FuC} = \frac{I}{z \cdot F} \tag{AE.11}$$

Since two electrons are released per hydrogen atom during oxidation at the anode, the valence $z = 2$. Consequently, equation AE.11 simplifies to:

$$\dot{n}_{H_2,FuC} = \frac{I}{2 \cdot F} \tag{AE.12}$$

The required hydrogen molar flow rate is proportional to the generated electrical current I and refers to a single fuel cell.

For calculating the hydrogen requirement of a fuel cell stack, the hydrogen molar flow rate $\dot{n}H_2, FuC$ required for one cell is multiplied by the number of cells Z in the stack. To ensure a constant hydrogen supply, hydrogen is supplied to the fuel cell in an over-stoichiometric ratio $\lambda Anode$. The resulting hydrogen requirement for the total system is calculated using the following equation:

$$\dot{n}_{H_2,System} = \frac{I}{2 \cdot F} \cdot Z \cdot \lambda_{Anode} \tag{AE.13}$$

If not all hydrogen in the fuel cell is converted to water through reaction with oxygen, the remaining hydrogen is removed from the fuel cell via a recirculation module and fed to the incoming hydrogen stream [226, p.23], [228, p.17].

Oxygen Supply

On the cathode side, oxygen supply is achieved through a compressor that compresses the air to the required pressure level. The goal is to make as much oxygen as possible available for the electrochemical reaction and to achieve a stoichiometric ratio of $\lambda \approx 2$ [201, p.97]. Since the pressure increase of the air simultaneously leads to a temperature increase, the air is subsequently cooled [228, p.17]. As with the hydrogen supply, the air humidity in the air stream is also increased to values between 70 and 80% through an air humidifier to minimize ohmic losses [201, p.96]. The oxygen requirement can be calculated analogously to the hydrogen requirement, noting that the valence of molecular oxygen is $z = 4$. Therefore, the oxygen molar flow rate is:

$$\dot{n}_{O_2,System} = \frac{I}{z \cdot F} \cdot Z \cdot \lambda_{Cathode} = \frac{I}{4 \cdot F} \cdot Z \cdot \lambda_{Cathode} \tag{AE.14}$$

Since it is assumed that oxygen is taken from the aircraft's environment to supply it to the fuel cell system, the oxygen concentration c_{O_2} must be considered when calculating the required air quantity. The lower the oxygen concentration, the more ambient air must be supplied to the fuel cell system [226, p.24]. Consequently, a required air quantity depending on the oxygen concentration results in:

$$\dot{n}_{Air,Sys} = \frac{I}{4 \cdot F} \cdot Z \cdot \lambda_{Cathode} \cdot \frac{1}{c_{O_2}} \tag{AE.15}$$

Through the electrochemical reaction, oxygen-depleted air is produced, which is discharged from the fuel cell via the cathode outlet. The oxygen concentration of the exhaust air can drop below 11%.

Power Electronics

The power electronics consist of four subsystems: power regulation, monitoring and control, and power supply management. The voltage supplied by the FuCS is highly dependent on the parameters of temperature, pressure, humidity, mass flow, and current, and changes according to the present load. According to O'Hayre et al. [203], most electronic devices require electrical energy that is delivered at a specific voltage level and remains stable over an extended period. For this reason, the voltage is regulated through DC converters. These convert the varying DC voltage generated by the fuel cell to a specified uniform voltage level, regardless of its input value [203, p.388]. Furthermore, during the operation of the fuel cell system, various variables, including stack temperature, gas mass flows, power output, and cooling, must be monitored and controlled. This task is solved through three separate subsystems:

1. System monitoring, where the fuel cell's state is monitored through measuring devices and sensors
2. System actuation, where individual control actuators (valves, pumps, or switches) are controlled to achieve a change in the state variable
3. The central control unit, which effects an interaction between the monitoring sensors and control actuators

Through these three subsystems, the fuel cell should be maintained in a stable and specified state, with most control systems using so-called feedback algorithms. Using O'Hayre et al.'s example, the feedback loop could be implemented between a temperature sensor and the thermal management subsystem. If the temperature of the fuel cell stack were

to rise, the control unit would increase the cooling airflow through the stack. Conversely, the control system would decrease the cooling airflow if the temperature of the fuel cell stack drops [203, p.368]. Fuel cells can exhibit slower dynamic response behavior than other electronic devices, such as batteries. This behavior can be attributed to the delay times of some system components, such as pumps or compressors. Therefore, batteries are integrated into the fuel cell system as buffer storage for faster system response [203, p.368f]. Power conditioning typically increases the investment costs of a FuCS by about 10-15%. Additionally, power conditioning causes a reduction in the electrical efficiency of the FuCS by approximately 5-20% [203]. Therefore, careful selection of the optimal power conditioning solution for a specific application is essential. Power regulation and power inversion are addressed next. The goal of regulation is to provide fuel cell power at a specific voltage level, since most applications require voltage to remain constant over time despite load variations. According to [203, p.388], fuel cell power is regulated with so-called DC converters.

Thermal Management System (TMS)

Since losses in the form of heat occur during the reaction in the fuel cell, an appropriate cooling system must ensure a constant operating temperature in the individual cells to prevent overheating and thus failure of the fuel cell. Fuel cells are, as described in A.14, galvanic cells that directly convert chemical energy into electrical energy. However, losses in the form of heat occur due to electrochemical reactions, ohmic resistance losses, and water condensation, which must be removed through appropriate cooling [197, pp.16-18], [205, p.3]. Two subfunctions are distinguished:

1. The transport of waste heat from the fuel cell (fuel cell cooling)
2. The removal of heat from the fuel cell system (system cooling)

Fuel Cell Cooling:
For the transport of waste heat from the fuel cell, two different cooling concepts are distinguished: Single-phase and two-phase cooling, which will be briefly explained below:

- Single-Phase Cooling: In this cooling method, see Figure AF.8, the coolant is circulated through the channels of the bipolar plates to absorb the waste heat and thus protect the fuel cell from overheating. In a heat exchanger located downstream of the fuel cell, the previously absorbed heat flow is released to the environment, causing the temperature of the cooling fluid to decrease. To compensate for varying fluid densities and leakages, a fluid reservoir is also necessary, see Figure AF.8. The coolant is selected according to specific criteria regarding viscosity, specific heat capacity, density, freezing point, electrical conductivity, and its corrosion properties, with Galden or a propylene glycol-water mixture (PGW) typically being used [197, p.19].
- Two-Phase Cooling: In this cooling concept, a pump delivers the liquid cooling fluid from a collection tank into the cooling channels of the fuel cell, where the cooling fluid evaporates due to the absorbed heat flow, see Figure AF.8. The now gaseous cooling fluid is then directed to the condenser, where it is liquefied using ambient air at a lower temperature level and flows back into the collection tank. The process can then begin anew. Ethanol, water, R245fa, and HFE-7100 are potential cooling fluids [197, p.20].

System Cooling:
The heat removed from the fuel cell through these two cooling concepts must, in the second step, be removed from the cooling circuit through a heat sink. Two different concepts are available for this:

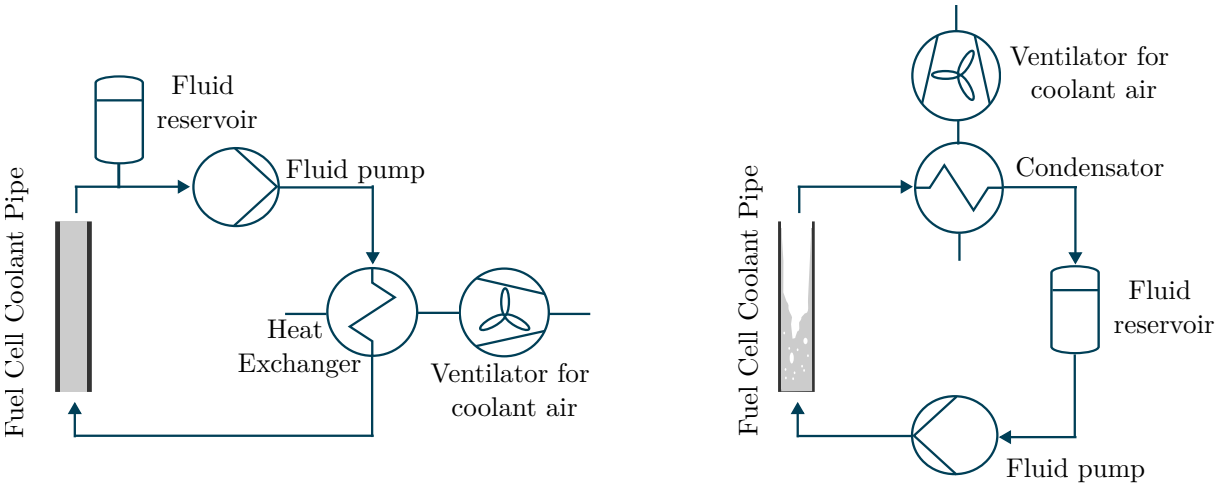

Figure AF.8.: Schematic system architecture of liquid cooling (left) and evaporative cooling (right), adapted from [199, p.15]

- **Heat Dissipation to the Environment**
  Compared to the second concept, this principle can remove the largest amounts of heat. The heat dissipation can occur either through a ram air duct or NACA inlet, or through the integration of a surface cooler into the aircraft skin [199, p.50]. In the first variant, the cool ambient air is drawn in through the ram air or NACA duct and fed to a heat exchanger or condenser that transfers the heat from the cooling circuit to the cool ambient air. With the integration of the surface cooler into the aircraft skin, however, the heat is transferred directly to the ambient air flowing around the aircraft. This has the great advantage that no additional air resistance is induced due to an additional inlet on the aircraft, but the disadvantage that the cooling capacity is highly dependent on the installation location and flight condition, resulting in a low cooling capacity/weight ratio [199, p.50].
- **Heat Dissipation into the Cabin**
  Under certain ambient conditions, part of the waste heat can be used for setting the desired cabin temperature. Particularly in the winter months, the heat could be used for heating the cabin. However, it should not be overlooked that the air conditioning system must be dimensioned larger to ensure a pleasant climate in the cabin even in summer months due to the additional heat flow.

A.15. Calculation of the Boundary Conditions for the Assumed FuCS Architecture

Note: The calculations and methodological approaches described here originate from a Master's thesis supervised by the author, which was adapted and translated into English for this dissertation. The original description can be found in Mr. Haidl's thesis [206].

To develop an effective TMS for fuel cell applications, a precisely defined reference system must be established. This section outlines the reference system architecture and boundary conditions that determine the operational parameters of the fuel cell system. The reference system model integrates literature data, theoretical calculations, and appropriate assumptions to create a foundation for the thermal analysis. This model encompasses:

- Environmental conditions across various flight phases
- Fuel cell stack performance specifications
- System architecture with focus on thermal interfaces

The objective is to determine the critical heat flows for dimensioning load cases, thereby establishing the interface parameters for the TMS.

Environmental Conditions

Flight conditions significantly impact the thermal management requirements of the fuel cell system. The analysis in this dissertation considers four primary flight phases with their distinct operational characteristics: taxi, climb, cruise, and holding. For each phase, the model accounts for altitude H, ambient temperature T, atmospheric pressure p, relative humidity RH, and Mach number Ma.

For cruise operations, an altitude of 11 km is assumed, consistent with typical operations for a commercial aircraft. The average altitude for climb phase is determined according to Scheiderer's methodology, while holding patterns follow the prescribed Munich Airport regulations [230].

The model incorporates International Standard Atmosphere (ISA) conditions (15°C, 1.01325 bar at sea level) as a baseline. To account for seasonal and geographical variations, additional scenarios for hot day (+15 K) and cold day (-15 K) operations are included in the analysis.

The temperature profile in the troposphere follows a linear gradient of approximately -0.65 K per 100 m. Following Scheiderer, the temperature t at a given altitude H can be calculated from ground temperature t_0 using:

$$T = T_0 - 0.0019812 \cdot H \tag{AE.16}$$

where T_0 is in degrees Celsius and H is expressed in feet.

The atmospheric pressure p is calculated by using the international altitude formula. Assuming a sea level as the reference height under ISA conditions, the equation takes the form:

$$p(H) = 1.01325 \left(1 - \frac{0.0065 \cdot H}{T_0 + 273.15}\right)^{5.255} \tag{AE.17}$$

where T_0 is given in °C and H is given in meters.

The relative humidity RH at ground level is assumed to be 75%, based on average annual values for Munich. At cruise altitude, the relative humidity decreases to approximately 20%, consistent with atmospheric measurements. For other flight phases, the relative humidity is adjusted according to the corresponding altitude.

The Mach number Ma is calculated as the ratio of flight speed v to the speed of sound c at the respective altitude:

$$Ma = \frac{v}{c} \tag{AE.18}$$

The speed of sound varies with altitude and is determined based on standard atmospheric tables. During taxi operations, the Mach number is zero, whereas in the cruise phase of a commercial aircraft like the A320, a typical Mach number is 0.78.

The environmental parameters for each flight phase have been systematically derived and are summarized in Table AT.35.

Table AT.35.: Environmental Parameters for Various Flight Conditions

Condition	**H [ft]**	**T [°C]**	**p [bar]**	**RH [%]**	**Ma [-]**
Taxi	0	15.0	1.013	75	0
Taxi Hot-Day	0	30.0	1.013	70	0
Taxi Cold-Day	0	0.0	1.013	80	0
Climb	5750	3.6	0.819	50	0.386
Climb Hot-Day	5750	18.6	0.828	45	0.376
Cruise	36089	-56.3	0.226	20	0.780
Holding	8000	-0.8	0.753	40	0.358

Fuel Cell Stack Model

The thermal management system design must account for the heat generation characteristics of the fuel cell stack across various operating conditions. For the reference system, the "P Stack" from PowerCellution was selected, which offers an advantageous power-to-weight ratio and compact form factor suitable for mobile applications [231].

While the stack provides a nominal maximum power of 125 kW, this has been theoretically scaled to 500 kW to meet the estimated power requirements of a medium-range commercial aircraft. It is important to note that such scaling introduces significant engineering challenges in real-world implementation. As Wang explains, non-uniform flow distribution within scaled stacks, resulting from suboptimal flow channel design, represents a major hurdle in translating theoretical models into practical applications. Effective scaling would require optimized flow and temperature conditions across all cells, which could be achieved through careful configuration of bipolar plates and appropriate flow channel dimensions.

The polarization curve, illustrated in Figure AF.9, illustrates how the stack voltage changes with increasing current density. This characteristic curve serves as the foundation for determining operating parameters at different power demands. The stack's performance varies significantly across flight phases, with each phase requiring a specific combination of cell count and operating points on the polarization curve.

The stack voltage U_{stack} is calculated from the individual cell voltage U_{cell} and the number of cells n_{cell} using:

$$U_{stack} = n_{cell} \cdot U_{cell} \tag{AE.19}$$

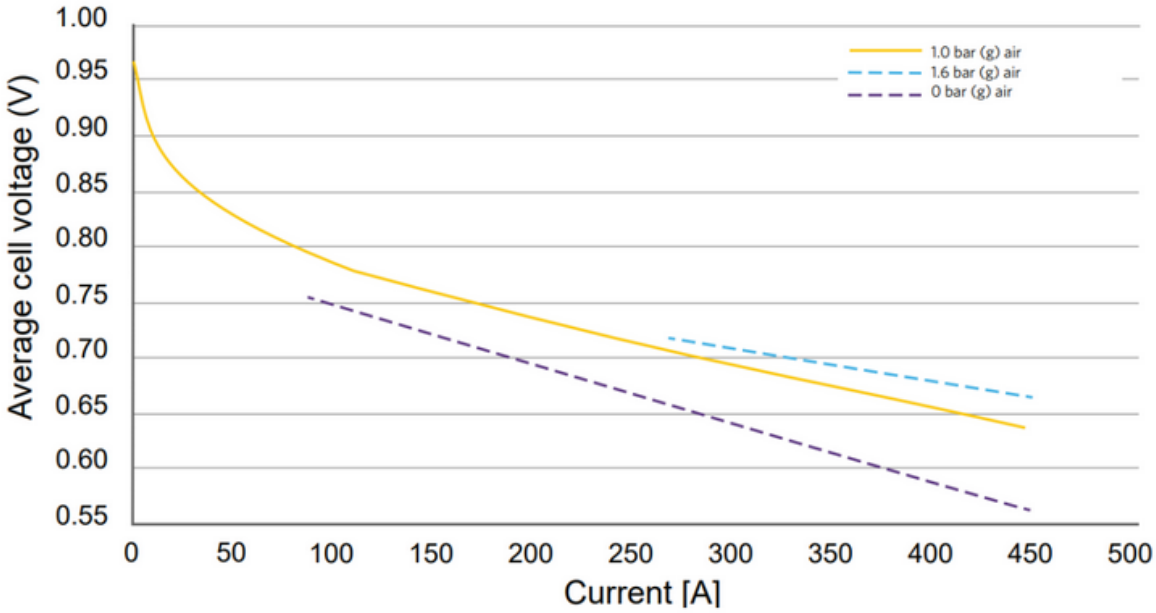

Figure AF.9.: Polarization curve of the fuel cell stack "P Stack" [231], adjusted.

Once the stack voltage is determined, the corresponding stack current I_{stack} can be calculated from the required power output P_{stack}:

$$I_{stack} = \frac{P_{stack}}{U_{stack}} \tag{AE.20}$$

These electrical parameters are essential for calculating reactant flow rates and associated thermal loads. The complete set of operational parameters for different flight phases is presented in Table AT.36, reflecting the varying power demands throughout the mission profile.

Table AT.36.: Fuel Cell Stack Parameters for Various Flight Phases

Parameter	Unit	Taxi	Climb	Cruise	Holding
Stack Power	[kW]	325	400	500	300
Cell Count	[-]	1183	1456	1820	1092
Cell Voltage	[V]	0.85	0.77	0.72	0.88
Stack Voltage	[V]	1006	1121	1310	961
Stack Current	[A]	323	357	382	312

Air Supply System

The air supply subsystem provides conditioned air to the fuel cell cathode with precise control of temperature, pressure, and humidity. For the reference system, the following parameters are assumed at the stack inlet:

- Air temperature: 70°C (allows for heating within the stack without exceeding the 80°C upper temperature limit)
- Air pressure: 2.0 bar (elevated pressure to reduce activation losses and facilitate humidification)
- Relative humidity: 90% (maintains membrane hydration without excessive condensation)

In addition, the following assumptions were made for the calculation of this subsystem:

- Calculation with total variables
- No pressure loss across the components and in the pipes
- Reversible adiabatic change of state in the air-side compressor
- Isobaric humidification and isobaric heat dissipation in the heat exchanger.

The system architecture incorporates a ram air duct with a NACA inlet, selected for its favorable aerodynamic characteristics and lower drag compared to conventional inlets. For ground operations when aircraft velocity is zero, an auxiliary fan is integrated to maintain the required air flow.

For the analysis, total (stagnation) properties of pressure and temperature are employed, which according to Weigand et al. can be calculated for an ideal gas as:

$$p_0 = p\left(1 + \frac{\kappa - 1}{2} \cdot Ma^2\right)^{\frac{\kappa}{\kappa-1}} \tag{AE.21}$$

$$T_0 = T\left(1 + \frac{\kappa - 1}{2} \cdot Ma^2\right) \tag{AE.22}$$

where the ratio of specific heat for air is $\kappa = 1.4$, p and T are static properties, and index 0 indicates total properties.

The stoichiometric oxygen mass flow $\dot{m}_{O_2,st}$ required by the electrochemical reaction is determined by:

$$\dot{m}_{O_2,st} = I_{stack} \cdot n_{cell} \cdot \frac{M_{O_2}}{4 \cdot F} \tag{AE.23}$$

Since ambient air rather than pure oxygen is used, the stoichiometric air mass flow $\dot{m}_{L,st}$ is calculated using the mass fraction of oxygen in air ($\xi_{O_2} = 23.14\%$):

$$\dot{m}_{L,st} = \frac{\dot{m}_{O_2,st}}{\xi_{O_2}} \tag{AE.24}$$

A stoichiometric ratio $\lambda_L = 2.0$ is applied, resulting in the actual air mass flow:

$$\dot{m}_L = \dot{m}_{L,st} \cdot \lambda_L \tag{AE.25}$$

For the isentropic compression process, the temperature relationship follows:

$$\frac{p_2}{p_1} = \left(\frac{T_2}{T_1}\right)^{\frac{\kappa}{\kappa-1}} \tag{AE.26}$$

The humidification system requires analyzing the thermodynamic states in an h-x diagram (Mollier diagram), where specific enthalpy of moist air h is plotted against moisture content x. The water content of unsaturated air is calculated as:

$$x = x_D = 0.622 \cdot \frac{p_D}{p - p_D} \tag{AE.27}$$

where p_D is the partial pressure of water vapor:

$$p_D = RH \cdot p_s \tag{AE.28}$$

The saturation pressure p_s is determined using the Antoine equation:

$$\log_{10}(p_s) = A - \frac{B}{T + C} \tag{AE.29}$$

with coefficients A, B, and C dependent on the temperature range. The specific enthalpy of unsaturated moist air is calculated as:

$$h = h_L + h_D = c_{p,L} \cdot t + x_D \cdot (c_{p,D} \cdot t + r_D) \tag{AE.30}$$

where $c_{p,L} = 1.01$ kJ/(kg·K) for air at 2 bar, $c_{p,D} = 2.2$ kJ/(kg·K) for water vapor, and $r_D = 2201$ kJ/kg for the latent heat of vaporization. For the steam injection process, the required steam mass flow is:

$$\dot{m}_D = \dot{m}_L \cdot (x_2 - x_1) \tag{AE.31}$$

The specific enthalpy of the steam is:

$$h_D = c_{p,D} \cdot t_D + r_D \tag{AE.32}$$

For the saturated steam temperature $t_D = 120.2$°C at 2 bar. The specific enthalpy after humidification is related to the steam enthalpy by:

$$h_D = \frac{h_2 - h_1}{x_2 - x_1} \tag{AE.33}$$

The temperature after humidification can be determined from:

$$h_2 = c_{p,L} \cdot t_2 + x_2 \cdot (c_{p,D} \cdot t_2 + r_D) \tag{AE.34}$$

Finally, the heat that must be removed in the subsequent heat exchanger is:

$$\dot{Q}_{HX_L} = \dot{m}_{ges} \cdot c_{p,G} \cdot \Delta T \tag{AE.35}$$

where the specific heat capacity of the gas mixture is:

$$c_{p,G} = c_{p,L} \cdot \frac{\dot{m}_L}{\dot{m}_{ges}} + c_{p,D} \cdot \frac{\dot{m}_D}{\dot{m}_{ges}} \tag{AE.36}$$

Hydrogen Supply System

The hydrogen supply subsystem must deliver appropriately conditioned hydrogen to the anode side of the stack with parameters complementary to those of the air system:

- Hydrogen temperature: 70°C (matching the air inlet temperature)

- Hydrogen pressure: 2.2 bar (slightly higher than cathode pressure)
- Relative humidity: 100% (optimal membrane hydration)

For aviation applications, hydrogen is stored cryogenically at approximately -252°C and 5 bar to achieve practical storage densities. This requires extensive thermal conditioning before use. The stoichiometric hydrogen mass flow is calculated according to Vredenborg's formula:

$$\dot{m}_{H_2,st} = I_{stack} \cdot n_{cell} \cdot \frac{M_{H_2}}{2 \cdot F} \tag{AE.37}$$

With a stoichiometric ratio $\lambda_{H_2} = 2.0$, the actual hydrogen mass flow becomes:

$$\dot{m}_{H_2} = \dot{m}_{H_2,st} \cdot \lambda_{H_2} \tag{AE.38}$$

The critical challenge in hydrogen processing is the extreme temperature differential between storage and operating conditions. The heat exchanger must provide sufficient energy to:

1. Heat liquid hydrogen from -252°C to its boiling point (approximately 23.2K at 2.2 bar)
2. Supply the latent heat for vaporization
3. Heat gaseous hydrogen to the required operating temperature

The total heat requirement is calculated as:

$$\dot{Q}_{HX_{H_2}} = \dot{m}_{H_2} \cdot (c_{p,H_2,fl} \cdot \Delta T_1 + r_{H_2} + c_{p,H_2,g} \cdot \Delta T_2) \tag{AE.39}$$

The specific heat capacities and vaporization enthalpy at 2.2 bar are determined from thermodynamic tables by linear interpolation:

- Specific heat capacity of liquid hydrogen: $c_{p,H_2,fl} = 11.73$ kJ/kg·K
- Specific heat capacity of gaseous hydrogen: $c_{p,H_2,g} = 13.71$ kJ/kg·K
- Specific vaporization enthalpy: $r_{H_2} = 428.3$ kJ/kg

The temperature difference ΔT_1 is calculated between the boiling point and the inlet temperature to the heat exchanger. For ΔT_2, the heat exchanger outlet temperature must be determined by working backward from the stack inlet:

$$T_{HX_{H_2},out} = T_{stack,in} - \Delta T_{hum+recirc} \tag{AE.40}$$

where $\Delta T_{hum+recirc}$ represents the temperature increase across the humidification and recirculation components. Due to the relatively small mass flows involved in these processes, a constant temperature increase of 10K is assumed regardless of load case.

In practical implementations, the large temperature range (over 300K) poses significant design challenges for conventional heat exchangers. One potential solution is to implement two heat exchangers in series, each optimized for a specific temperature range, though this would increase system weight and complexity.

Heat Flow Analysis

The thermal management system must be dimensioned to handle heat flows from multiple sources and to various sinks within the fuel cell system architecture. A comprehensive heat flow analysis is essential for identifying the critical dimensioning load cases.

The primary heat sources in the system include:

- Fuel cell stack waste heat
- DC/DC converter losses
- Air conditioning heat exchanger (during cooling operations)

The corresponding heat sinks are:

- Hydrogen conditioning heat exchanger
- Ram air heat exchanger

The fuel cell electrochemical processes are inherently inefficient, with considerable energy converted to heat rather than electricity. Based on established literature and experimental data, a thermal efficiency of approximately 50% for the PEMFC stack is assumed. Consequently, for each kilowatt of electrical output produced, roughly an equal amount of thermal energy is generated.

However, not all of this thermal energy requires active cooling through the thermal management system. According to experimental studies by Fly and Thring [232, p.274], approximately 16% of the energy from the stack is carried away by the exhaust gases. Berger's research indicates that between 5-10% of the total energy input is removed as thermal energy in the exhaust [225, p.16]. For this analysis, a conservative value of 12% for exhaust heat removal is adopted, independent of the operating point. The heat that must be actively managed by the cooling system $\dot{Q}_{HX_{BZ}}$ can therefore be calculated as:

$$\dot{Q}_{HX_{BZ}} = (1 - \eta_{exh}) \cdot P_{stack} \tag{AE.41}$$

where $\eta_{exh} = 0.12$ represents the fraction of heat removed by the exhaust, and P_{stack} is the electrical power output of the stack.

The DC/DC converter, which conditions the electrical output to meet aircraft power system requirements, introduces additional thermal load. With an efficiency $\eta_{DC} = 0.95$, the heat generation from this component is calculated as:

$$\dot{Q}_{HX_{DC}} = (1 - \eta_{DC}) \cdot P_{stack} \tag{AE.42}$$

To determine the heat that must be rejected through the ram air heat exchanger $\dot{Q}_{HX_{SL}}$, a heat flow balance across all sources and sinks is established:

$$\dot{Q}_{HX_{BZ}} + \dot{Q}_{HX_{DC}} + \dot{Q}_{HX_L} = \dot{Q}_{HX_{H_2}} + \dot{Q}_{HX_{SL}} \tag{AE.43}$$

Rearranging to solve for the ram air heat exchanger load:

$$\dot{Q}_{HX_{SL}} = \dot{Q}_{HX_{BZ}} + \dot{Q}_{HX_{DC}} + \dot{Q}_{HX_L} - \dot{Q}_{HX_{H_2}} \tag{AE.44}$$

Table AT.37 presents the comprehensive heat flow analysis for all considered operating conditions. The data reveals that cruise conditions generate the highest thermal load (433.2 kW) that must be rejected through the ram air heat exchanger, primarily due to the maximum stack power in this phase. However, the "Taxi Hot-Day" scenario presents unique challenges due to the high ambient temperature, which reduces the temperature differential available for heat rejection despite the lower power demand.

Table AT.37.: Heat Flows [kW] for Various Flight Phases

Flight Phase	Heat Sources			Heat Sinks	
	FuC Stack	**DC/DC HX**	**Air HX**	**H_2 HX**	**Ram Air HX**
Taxi	286.0	16.3	4.6	37.5	269.3
Taxi Hot-Day	286.0	16.3	9.0	37.5	273.7
Taxi Cold-Day	286.0	16.3	0.2	37.5	264.5
Climb	352.0	20.0	9.0	51.0	330.0
Climb Hot-Day	352.0	20.0	14.1	51.0	335.1
Cruise	440.0	25.0	36.3	68.1	433.2
Holding	264.0	15.0	6.7	33.5	252.3

Thermal Management System Interface Parameters

The thermal management system interfaces with multiple subsystems within the fuel cell architecture, each with distinct temperature, pressure, and flow requirements. These interface parameters define the boundary conditions for thermal management system design and must be precisely controlled to ensure optimal system performance.

For the internal cooling loop of the fuel cell stack, a water-glycol mixture (50:50 ratio) is specified as the coolant medium. This formulation from BASF offers enhanced corrosion and frost protection compared to pure water, while maintaining low electrical conductivityan essential property for fuel cell cooling applications. The interface parameters for this loop are:

- Inlet temperature: 80°C (operating at the upper boundary of the PEM fuel cell temperature range)
- Minimum outlet temperature: 70°C (limiting temperature differential to 10 K to prevent thermal stress)
- Inlet pressure: 1.2 bar (maintaining positive pressure differential relative to ambient to prevent gas ingress)

The maximum temperature differential constraint of 10K is particularly important for preserving stack durability, as larger thermal gradients can induce mechanical stresses in the materials and accelerate degradation.

For the DC/DC converter cooling interface, similar temperature and pressure parameters are adopted to facilitate potential integration of cooling loops. The mass flow rates for each component are derived from the heat load analysis and the specified temperature constraints using the energy balance equation:

$$\dot{m} = \frac{\dot{Q}}{c_p \cdot \Delta T} \tag{AE.45}$$

where $\dot{m}$ is the mass flow rate, $\dot{Q}$ is the heat load, c_p is the specific heat capacity of the coolant at constant pressure (evaluated at the mean temperature), and ΔT is the allowable temperature rise.

For the interfaces with the air conditioning and hydrogen conditioning subsystems, the mass flow rates are load-dependent and calculated based on the stoichiometric requirements previously described, while the interface with the ram air heat exchanger depends on flight phase and ambient conditions.

Table AT.38 presents the complete set of interface parameters for the two most challenging operating conditions: cruise (highest power demand) and taxi on a hot day (most challenging ambient conditions). These parameters establish the foundational requirements for TMSm architecture development and component sizing. These interface parameters provide the foundation for developing TMS architectures that can effectively maintain appropriate operating conditions for all components across the entire flight envelope.

Table AT.38.: TMS Interface Parameters

Component	**Parameter**	**Unit**	**Cruise**	**Taxi Hot-Day**
FC Stack	Inlet Temperature	[°C]	80	80
	Outlet Temperature	[°C]	70	70
	Inlet Pressure	[bar]	1.2	1.2
	Mass Flow	[kg/s]	12.29	12.29
DC/DC Converter	Inlet Temperature	[°C]	80	80
	Outlet Temperature	[°C]	70	70
	Inlet Pressure	[bar]	1.2	1.2
	Mass Flow	[kg/s]	0.698	0.698
Air Conditioning	Inlet Temperature	[°C]	129.15	97.18
	Outlet Temperature	[°C]	70	70
	Inlet Pressure	[bar]	2.0	2.0
	Mass Flow	[kg/s]	0.548	0.298
	Inlet RH	[%]	10.5	30
	Outlet RH	[%]	90	90
Hydrogen Conditioning	Inlet Temperature	[°C]	-252	-252
	Outlet Temperature	[°C]	60	60
	Inlet Pressure	[bar]	2.2	2.2
	Mass Flow	[kg/s]	0.0145	0.008
Ram Air	Inlet Temperature	[°C]	-29.9	30
	Inlet Pressure	[bar]	0.336	1.01
	Inlet RH	[%]	20	70

Input Parameters for the Power Calculation of the FuCS (see page 257)

Table AT.39.: Assumed mission data

	H [ft]	**H [m]**	**v_{flight} [kt]**	**v_{flight} [m/s]**	**Speed of Sound [m/s]**	**Ma**	**$T_{ambient}$ [°C]**	**$T_{ambient}$ [K]**	**$p_{ambient}$ [bar]**
Taxi-Out									
Taxi	0	0	15	7.71	340.5	0.02	15	288.15	1.01325
Take-Off									
Acceleration	35	10.67	150	77.1	340.46	0.23	14.93	288.08	1.012
Climb									
Initial Climb	4,000	1,219.2	190	97.66	335.75	0.29	7.08	280.23	0.88
Climb 1	15,000	4,572	290	149.06	322.67	0.46	-14.72	258.43	0.572
Climb 2	24,000	7,315.2	350	179.9	311.97	0.58	-32.55	240.60	0.393
Climb 3	40,000	12,192	410	210.74	292.95	0.72	-56.5	208.90	0.187
Cruise									
Cruise	40,000	12,192	468.98	241.07	297.6	0.81	-56.5	216.65	0.187
Descent									
Descent 1	30,000	9,144	480.39	246.92	304.84	0.81	-44.44	228.72	0.3009
Descent 2	10,000	3,048	250	128.5	328.62	0.391	-4.81	268.3	0.697
Descent 3	4,000	1,219.2	240	123.36	335.75	0.367	7.08	280.23	0.875
Approach									
Approach Initial	2,500	762	240	123.36	337.53	0.365	10.047	283.197	0.925
Approach Final	1,000	304,8	160	82.24	339.31	0.242	13.018	286.169	0.977
Landing									
Thresh. Cros.	100	30.48	140	71.96	340.38	0.212	14.802	287.952	1.0095
Touchddown	0	0	137	70.418	340.5	0.207	15	288.15	1.01325
Deceleration	0	0	15	7.71	340.5	0.023	15	288.15	1.01325
Taxi-In									
Taxi	0	0	15	7.71	340.5	0.023	15	288.15	1.01325

A.16. High-Level Questions for the TMS Development

ID	Question
System Context - Focus on boundaries, interfaces, etc.	
SC.1	What are the physical boundaries of thermal management systems that interface with other systems and the environment?
SC.2	What are the specific operational envelope parameters for TMS design including temperature ranges, thermal loads, environmental conditions, and performance constraints?
SC.3	Which stakeholders (equipment operators, maintenance personnel, system integrators, regulatory entities) interact with the TMS throughout its lifecycle, and what are their specific requirements and touchpoints with the system?
SC.4	What international standards, manufacturer specifications, and regulatory documents explicitly define the requirements, interface specifications, and operational parameters for modern thermal management systems?
SC.5	What energy sources, coolants, refrigerants, and other consumables are required for thermal management system operation in different applications?
Requirements - Focus on quantifiable specifications and constraints	
RQ.1	What are the quantitative performance requirements for temperature control, including setpoint accuracy (±°C), spatial temperature variation limits, and responsiveness metrics (time to achieve setpoint) across different operational conditions?
RQ.2	What specific reliability requirements are defined for TMS, including redundancy levels, and maximum allowable failure rates for different system functions?
RQ.3	What are typical environmental and operational constraints (size, weight, power consumption, noise levels, vibration resistance) that apply to thermal management systems?
RQ.4	What specific safety requirements and thresholds govern TMS design, especially regarding coolants, pressure vessels, and electrical components?
RQ.5	What verification and validation methods, including specific test procedures, simulation requirements, and acceptance criteria, are standard for TMS certification across different industries?
Functions - Focus on capabilities and operational behavior	
FU.1	What are the core functions of thermal management systems across various applications?
FU.2	What sensing, measurement, and monitoring functions are implemented in thermal management systems and what parameters do they track?
FU.3	What functions allow the TMS to transition between different operational modes (startup, normal operation, high load, low power, shutdown, emergency)?
FU.4	What safety and protection functions prevent thermal management system failures or other unsafe conditions in operation?
FU.5	What diagnostic and self-monitoring functions exist within modern TMS designs?

ID	Question
Logical Elements - Focus on system decomposition and information flow	
LE.1	How are thermal management systems decomposed into functional subsystems and what are their primary responsibilities?
LE.2	What are the coolant flow, thermal energy, and control signal paths between thermal management subsystems and how are they coordinated?
LE.3	What control architectures (centralized, distributed, redundant) are used in thermal management systems and how is control logic implemented?
LE.4	What is the complete information flow architecture within the TMS, including all sensor inputs, control signals, actuation commands, and feedback loops with specific signal types and communication protocols?
LE.5	What are the critical dependencies between thermal management system components and other systems that affect overall thermal performance?
Physical Components - Focus on hardware implementation	
PC.1	What are the essential physical components of thermal management systems and their typical specifications?
PC.2	What specific heat exchanger designs and cooling technologies are used in modern TMS applications?
PC.3	What materials are used for critical thermal management system components and what properties (thermal conductivity, specific heat capacity, durability) make them suitable for specific applications?
PC.4	What pump, valve, and flow control technologies are employed throughout the TMS, including specific actuation methods, flow capacities, response times, and failure modes?
PC.5	What commercial-off-the-shelf (COTS) components versus custom components are typically used in thermal management systems for different applications and industries?
Architecture Patterns - Focus on design patterns and system structure	
AP.1	What reference architectures and design patterns are commonly used for thermal management systems in different applications (electronics cooling, vehicle thermal management, power systems cooling)?
AP.2	How is modularity achieved in thermal management systems to allow for maintenance, scaling, and adaptation to different thermal loads?
AP.3	What redundancy strategies are implemented to ensure reliability in critical thermal management systems?
AP.4	How are interfaces standardized within thermal management systems to enable component interchangeability and system integration across platforms?
AP.5	What architectural trade-offs impact thermal management system performance, efficiency, size, weight, and cost in different design scenarios?

A.17. Representative Excerpts from TMS Knowledge Synthesis Report

The TMS documentation has been systematically analyzed and synthesized from 115 literature sources. This appendix presents the essential tabular documentation extracted from the TMS report. These tables provide a foundational reference for TMS architecture design and cross-domain applicability assessment.

Note: All content directly reproduced from the original LLM-generated report is highlighted in light gray. This convention applies uniformly to text passages, lists, and tables.

1. Requirements Analysis
This document synthesizes universal requirements for Thermal Management Systems (TMS) in PEM fuel cell applications across aircraft and related domains, extracted from a comprehensive literature database. It covers functional, thermal control, safety, reliability, integration, and control requirements, emphasizing technology-agnostic, scale-independent principles. Quantitative specifications and design trade-offs are highlighted, with traceability to standards such as SAE ARP4761 and ARP4754A. The analysis supports informed design decisions for robust, efficient, and safe TMS architectures.

Table AT.40.: TMS synthesis report - functional requirements

Requirement ID	Description	Critic. Level	QA-pair ID
REQ-FUNC-001	Regulate temperature of critical components (fuel cells, batteries, motors) to maintain safe and efficient operation.	High	[FU.1]
REQ-FUNC-002	Extract and transfer heat from components to heat sinks or environment via heat exchangers and radiators.	High	[FU.1]
REQ-FUNC-003	Maintain coolant quality by removing particles and deionizing to prevent clogging and electrical shorts.	High	[FU.1]
REQ-FUNC-004	Coordinate thermal management across multiple subsystems to improve compactness, energy efficiency, and temperature control.	High	[FU.1]
REQ-FUNC-005	Provide controlled heating to components during cold conditions to maintain operational temperature limits.	Medium	[FU.1]
REQ-FUNC-006	Continuously measure temperature, pressure, and flow parameters and employ control strategies to maintain thermal stability and uniformity.	High	[FU.1]

2. Functional Analysis
This report presents an exhaustive extraction and categorization of all functions identified in the literature for Thermal Management Systems (TMS) in aircraft and related applications. The analysis is technology-agnostic and scale-independent, covering core functional capabilities, control strategies, safety and reliability mechanisms, integration considerations, and component specifications. Each function is precisely defined, with traceability to direct literature quotes and QA-pair identifiers. The report further assesses cross-domain applicability, hierarchical dependencies, and integration challenges, providing a foundational registry to support system architecture design, verification, and future research.

Table AT.41.: TMS synthesis report - comprehensive function registry

Func. ID	Function Name	Function Description	Func. Categ.	Critic. Level	Related Components	QA ID
FUNC-001	Control Temperature	Maintain temperature of aircraft systems and cabin environment within acceptable operational and comfort limits	Primary	High	Heat Exchangers, Pumps, Sensors, Control Units	FU.1
FUNC-002	Acquire Heat	Extract heat from diverse sources including engines, avionics, electrical equipment, hydraulic systems, and occupants	Primary	High	Heat Sources, Heat Exchangers	SC.1
FUNC-003	Transport Thermal Energy	Transfer thermal energy via fluids, heat exchangers, ducts, pumps, and control equipment to heat sinks	Primary	High	Heat Exchangers, Pumps, Ducts, Valves	SC.1
FUNC-004	Reject Heat	Safely dissipate heat to terminal heat sinks such as ambient air or fuel	Primary	High	Heat Sinks, Heat Exchangers, Fans	SC.1
FUNC-005	Maintain Coolant Quality	Remove particles and deionize coolant to prevent clogging and electrical shorts	Support	High	Filters, Deionization Units	FU.1
FUNC-006	Provide Control. Heating	Heat components during cold conditions to maintain operational temperature limits	Support	Medium	Heaters, Control Units	FU.1
FUNC-007	Monitor Thermal Parameters	Continuously measure temperature, pressure, flow, and fluid quality to enable adaptive control	Support	High	Sensors (Temp., Pressure, Flow), Control Units	FU.1
FUNC-008	Control Coolant Flow	Regulate pump speeds and valve positions to maintain desired coolant flow rates and temperatures	Primary	High	Pumps, Valves, Control Units	FU.2.2.1
FUNC-009	Manage Thermal Load Distribution	Coordinate multiple cooling loops and heat exchangers to balance thermal loads and maintain system stabil.	Primary	High	Cooling Loops, Bypass Valves, Pumps	PC.1.2.2
FUNC-010	Detect & Isolate Faults	Implement fault detection and isolation mechanisms using model-based observers and active fault-tolerant control	Support	High	Sensors, Control Units, Fault Detection Algorithm	FU.5
FUNC-011	Provide Load Shedding	Reduce or shed non-essential electrical loads during thermal overload or failure conditions	Support	High	Control Units, Electrical Load Management	FU.3.1
FUNC-012	Announce Alerts	Provide timely, prioritized, and unambiguous alerts to flight crew for thermal management system status and failures	Support	High	Alerting Systems, Control Units	FU.4.2.2

2a. Function Categorization Framework

- Primary Functions: Core capabilities essential for TMS operation, including temperature control, heat acquisition, thermal transport, heat rejection, and coolant quality maintenance.
- Support Functions: Functions enabling or enhancing primary functions, such as controlled heating, monitoring, fault detection, load shedding, and alerting.
- Control Functions: Algorithms and control logic managing actuators (pumps, valves, fans) based on sensor feedback to maintain thermal stability and optimize energy efficiency.
- Safety Functions: Fail-safe mechanisms, redundancy, fault tolerance, and safety-critical alerting ensuring system reliability and compliance with certification standards.
- Integration Functions: Interface management, resource routing, and coordination among subsystems (ECS, FuCS, PTMS) to ensure cohesive system operation.

Table AT.42.: TMS synthesis report - function hierarchy and dependencies analysis

Function ID	Depends On	Enables	Dependency Type	Criticality Impact
FUNC-001	FUNC-007, FUNC-008	FUNC-011, FUNC-012	Sequential	Critical Path
FUNC-008	FUNC-007	FUNC-001	Feedback Loop	High
FUNC-010	FUNC-007	FUNC-012	Safety-Critical	High
FUNC-011	FUNC-001	System Stability	Conditional	High
FUNC-003	FUNC-002	FUNC-004	Sequential	Critical Path
FUNC-009	FUNC-003, FUNC-008	FUNC-001	Coordination	High

Table AT.43.: TMS synthesis report - cross-domain function applicability matrix

Function ID	Automotive	Aerospace	Stationary	Marine	Protable	Notes on Variations
FUNC-001	Yes	Yes	Yes	Yes	Yes	Universal function with scale-dependent implementation
FUNC-008	Yes	Yes	Yes	Yes	Yes	Control algorithms adapted to domain-specific dynamics
FUNC-010	Yes	Yes	Yes	Yes	Yes	Model-based fault detection widely applicable
FUNC-011	Yes	Yes	No	Yes	No	Load shedding strategies vary by domain and system criticality
FUNC-012	Yes	Yes	Yes	Yes	Yes	Standardized alerting principles across domains

2b. Functional Gap and Conflict Analysis

- No explicit numeric pressure drop limits for heat exchangers and coolant loops were found, indicating a gap in standardized design criteria.

- Limited detailed specifications for sensor response times and sampling rates, suggesting a need for further research on sensor integration in TMS.
- Variations in load shedding thresholds and control strategies across domains require harmonization for cross-domain applicability.
- Conflicting definitions of thermal safety margins exist, with some sources suggesting ± 5% while others use ± 15%, necessitating standardization.

3. Comprehensive Logical Elements Analysis

Table AT.44.: TMS synthesis report - logical elements registry

Element ID	Element Name	Element Description	Associated Functions	Interface Req.	QA ID
LE-001	Heat Exchanger	Device that transfers heat between fluids without mixing, designed to be lightweight, compact, and efficient, often with custom geometries to fit aircraft constraints	FUNC-002, FUNC-003, FUNC-004	Fluid interfaces (inlet/outlet), thermal interfaces, pressure ratings	PC.1
LE-002	Coolant Pump	Mechanical or electrical device that circulates coolant through thermal loops, sized to meet flow and pressure requirements while minimizing power consumption and weight	FUNC-003, FUNC-008	Electrical and mechanical interfaces, flow rate and pressure control	PC.4
LE-003	Control Valve	Actuated device regulating fluid flow paths and rates within thermal loops, enabling dynamic control of coolant routing and heat rejection	FUNC-003, FUNC-008	Electrical actuation signals, flow control interfaces	PC.4
LE-004	Temp. Sensor	Device measuring temperature at critical points to provide accurate, timely data for thermal control and fault detection	FUNC-007, FUNC-010	Electr. signal outputs, environmental robustness	FU.2.1.1
LE-005	Thermal Control Unit	Computational unit implementing control algorithms (e.g., PID, MPC, ADRC) to regulate actuators based on sensor feedback	FUNC-008, FUNC-009, FUNC-010	Sensor inputs, actuator command outputs, communication interfaces	PC.4.2
LE-006	Heat Pipe	Passive thermal transport device using phase change and capillary action to transfer heat efficiently without moving parts	FUNC-003	Thermal interface, fluid compatibility	PC.2
LE-007	Load Shedding Function	Function to reduce or disconnect non-essential electrical loads during thermal overload or failure conditions	FUNC-011	Control signals to power management systems, pilot interfaces	FU.3.1
LE-008	Alerting System	System providing prioritized, unambiguous alerts to flight crew for thermal management system status and failures	FUNC-012	Visual, aural, tactile alert interfaces	FU.4.2.2

Table AT.45.: TMS synthesis report - traceability matrix - requirements to elements

Requirement ID	Element ID	Implementation Variant	Test Procedure	Verification Status
REQ-FUNC-001	LE-001	Plate-fin, microchannel	Thermal perf. testing, CFD validation	Verified
REQ-FUNC-005	LE-004	RTD sensors, thermocouples	Accuracy calibration, environmental testing	Verified
REQ-FUNC-007	LE-007	Hierarchical load shedding control	System integration testing, safety assessment	Verified
REQ-FUNC-008	LE-008	Logic-based multi-sensory alerting	Flight crew alerting system validation	Verified

4. Comprehensive Component Invenctory and Analysis

Table AT.46.: TMS synthesis report - comprehensive component registry (excerpt)

Component ID	Compon. Name	Component Description	Critical Necess.	Physical Specifications	QA-ID
COMP-001	Heat Exchanger	Device transferring heat between fluids without mixing, designed to be lightweight, compact, & efficient	Very High	Custom geometries; materials: aluminium, composites	PC.1
COMP-002	Coolant Pump	Mechanical or electrical device circulating coolant through thermal loops	Very High	Flow rates: 5-150 LPM; power: 1.5 - 5 kW	PC.4
COMP-003	Control Valve	Actuated device regulating fluid flow paths and rates within thermal loops	Very High	Electrical actuation; fast response	PC.4
COMP-004	Temp. Sensor	Device measuring temperature at critical points to provide accurate, timely data	Very High	Accuracy ±0.5°C; fast response	FU.2.1.1
COMP-005	Thermal Control Unit	Computational unit implementing control algorithms to regulate actuators	Very High	Real-time control; fault-tolerant; adaptive	PC.4.2
COMP-006	Heat Pipe	Passive thermal transport device using phase change and capillary action	High	Passive; high thermal conductivity; compact	PC.2
COMP-007	Load Shedding Module	Function to reduce non-essential electrical loads during thermal overload conditions	High	Hierarchical control; power management integration	FU.3.1
COMP-008	Alerting System	System providing prioritized alerts to flight crew for thermal management status	High	Logic-based; multi-sensory cues; fault-tolerant	FU.4.2.2

A.18. Representative Excerpts from TMS Architecture Generation Report

This appendix presents representative excerpts from the TMS architecture generation report. They illustrate the LLM-based synthesis process and serve as reference examples of the generated TMS architecture. The color highlights indicate the origin of each element: literature-derived knowledge (green), synthesized system engineering elements (orange), and aircraft-specific data or assumptions (red). All contents are reproduced verbatim from the original report.

Note: All content directly reproduced from the original LLM-generated report is highlighted in light gray. This convention applies uniformly to text passages, lists, and tables.

1. Requirements Analysis and Derivation

Table AT.47.: TMS generation report - functional requirements extraction

Requir. ID	Description	Source	Critic.	Verification Method	Mapped Funct.
REQ-TMS-001	Regulate the temperature of critical components (fuel cells, batteries, motors) to maintain safe and efficient operation	[FU.1], Prov. HTML	High	Validat. test under ground and flight conditions	FUNC-TMS-01, FUNC-TMS-02
REQ-TMS-002	Extract and transfer heat from components to heat sinks or environment via heat exchangers and radiators	[FU.1], Prov. HTML	High	Lab heat exchanger performance test	FUNC-TMS-01, FUNC-TMS-03
REQ-TMS-003	Maintain coolant quality by removing particles and deionizing to prevent clogging and electrical shorts	[FU.1], Prov. HTML	High	Filter and deionizer subsystem test	FUNC-TMS-04
REQ-TMS-004	Manage thermal loads at up to 440 kW (cruise) for the FC Stack HX, ensuring coolant inlet <70°C	Aircraft-Specific	High	Flight envelope testing, simulation	FUNC-TMS-01, FUNC-TMS-05
REQ-TMS-005	Continuously measure temperature, pressure, and flow parameters and employ control strategies to maintain thermal stability and uniformity	[FU.1], Prov. HTML	High	Full hardware-in-loop testing	FUNC-TMS-03, FUNC-TMS-06
REQ-TMS-006	Provide controlled heating to components during cold conditions to maintain operational temperature limits	[FU.1], Prov. HTML	Medium	Low-temp. ground test	FUNC-TMS-07
REQ-TMS-007	Use model-based observers and active fault-tolerant control (e.g., UKF and SMC) to handle sensor and actuator faults	[SC.3], Prov. HTML	High	Fault injection simulat., flight test	FUNC-TMS-08
REQ-TMS-008	Ensure compliance with SAE ARP4754A, ARP4761 across system design and safety analysis	Synthesized from literature & certific. specs	High	Certification documentation & audits	FUNC-TMS-09

Table AT.48.: TMS generation report - performance requirements for aircraft case

Parameter	Specification	Criticality	Reference
Temp. Control Accuracy	±0.5°C	High	[SC.3.1], [SC.3.2.1] - Literature
Heat Transfer Capacity	Max 440 kW (FC stack) + up to 36.3 kW (AC), etc., with margin	High	Derived from flight-phase data
Response Time for Load Changes	Within 50 seconds with SMC-based FTC	High	[SC.3.1], [SC.3.2] - Literature
Pressure Drop Limit	<0.5 bar across entire TMS loop (Assumption)	High	Aircraft-Specific Gap (No universal standard found)
Coolant Flow Rate	5 - 150 L/min adjustable depending on phase	Medium	Based on typical pump design in [PC.4]

2. Compehensive Function Analysis

Table AT.49.: TMS generation report - function registry for aircraft application

Function ID	Function Name	Description	Source	Performance Requirements
FUNC-TMS-01	Temperature Control	Maintain the temperature of aircraft systems and cabin environment within acceptable operational and comfort limits	[FUNC-001], Literature	±0.5°C accuracy, Coolant inlet <70°C for FC
FUNC-TMS-02	Heat Acquisition	Extract heat from diverse sources including engines, avionics, electrical equipment, hydraulic systems, and occupants	[SC.1], Literature	Efficient HX usage across 286-440 kW range
FUNC-TMS-03	Thermal Transport	Transfer thermal energy via fluids, heat exchangers, ducts, pumps, and control equipment to heat sinks	[SC.1], Literature	Minimal pressure drop <0.5 bar, Flow rate 5-150 L/min
FUNC-TMS-04	Coolant Quality Maintenance	Remove particles and deionize coolant to prevent clogging and electrical shorts	[FU.1], Literature	Conductivity <3 μS/cm
FUNC-TMS-05	Heat Rejection	Safely dissipate heat to terminal heat sinks such as ambient air or fuel	[SC.1], Literature	Ram air cap. up to ~430 kW rejection
FUNC-TMS-06	Monitoring and Control	Continuously measure temperature, pressure, and flow parameters and employ control strategies	[FU.1], Literature	Temperature set-point ±0.5°C, quick response <50s
FUNC-TMS-07	Controlled Heating	Provide controlled heating to components during cold conditions to maintain operational temp. limits	[FU.1], Literature	Functional from -20°C ambient
FUNC-TMS-08	Fault Detection & Isolation	Implement fault detection using model-based observers (UKF) and active fault-tolerant control	[FU.5], Literature	99% fault detection accuracy, minimal false positives
FUNC-TMS-09	Safety & Certification Management	Ensure compliance with SAE ARP4761, ARP4754A, incorporate FMEA, redund., & safety controls	Synthesized standards	Full coverage in design & validation phases

Table AT.50.: TMS generation report - function hierarchy for aircraft application

Function ID	Depends On	Enables	Dependency Type
FUNC-TMS-01 (Temperature Control)	FUNC-TMS-06 (Monitoring), FUNC-TMS-03 (Transport)	Overall TMS operation	Core / Parallel
FUNC-TMS-02 (Heat Acquisition)	N/A	FUNC-TMS-03 (Transport)	Sequential
FUNC-TMS-05 (Heat Rejection)	FUNC-TMS-03 (Transport)	System Cooling	Sequential
FUNC-TMS-08 (Fault Detection)	FUNC-TMS-06 (Monitoring)	System Safety & Alerts	Safety-Critical

3. Logical Elements Analysis

The TMS is comprised of physical and control-oriented "logical elements" implementing the required functions:

Table AT.51.: TMS generation report - logical elements registry for aircraft application

ID LE-TMS	Element Name	Description	Implemented Functions	Physical Implementation
-01	Heat Exchanger Network	"Device that transfers heat between fluids without mixing, designed to be lightweight, compact, and efficient"	FUNC-TMS-02 (Heat Acquisition), FUNC-TMS-05 (Heat Rejection)	Plate-fin, microchannel, or advanced composite HX
-02	Coolant Pump & Valve Module	"Mechanical or electrical device that circulates coolant ... Valves regulate fluid flow path"	FUNC-TMS-03 (Transport), FUNC-TMS-06 (Monitoring & Control Support)	Axial piston pump or elec.pump with electro-hydraulic valves
-03	Sensor Suite	"Temperature sensors placed at critical points with ±0.5°C accuracy. Includes pressure & flow sensors"	FUNC-TMS-06 (Monitoring), FUNC-TMS-08 (Fault Det. Input)	RTDs, thermocouples, flow meters, pressure transducers
-04	Controller & Fault-Tolerant Control Unit	"Control units employing advanced control algorithms such as MPC, ADRC, and PID plus model-based observers (UKF)"	FUNC-TMS-01 (Temp. Control), FUNC-TMS-08 (Fault Detection), FUNC-TMS-09 (Safety Manag.)	Digital signal processor or embedded CPU with ARINC or CAN interfaces
-05	Coolant Conditioning Module	"Maintain coolant quality by removing particles and deionizing to prevent clogging and shorts"	FUNC-TMS-04 (Coolant Quality Maintenance)	Replaceable filter cartridges, ion-exchange resin, integrated sensor
-06	Heating Element/Cold-Start Subsystem	"Provide controlled heating to components during cold conditions to maintain operational temperature limits"	FUNC-TMS-07 (Controlled Heating)	Electric heater coil or small fluid-based heater circuit

4. Architecture Concept and Initial Design Decisions

Choice: Single-Phase vs. Two-Phase Cooling

We adopt a single-phase cooling system using a 50/50 water-glycol mixture because it provides a proven, reliable means of thermal control with moderate complexity, good heat capacity, and freeze protection. Two-phase systems often achieve higher heat transfer coefficients but introduce greater system complexity, potential for flow instabilities, and stricter control requirements. For this aircraft application, reliability and simplicity are prioritized over the potential weight savings of a two-phase approach.

Key Design Principles:

- Redundancy for critical loads (e.g., additional pump or parallel lines for the FC stack HX).
- Fail-safe operation with bypass valves and relief routes to handle off-nominal conditions.
- Weight optimization through compact heat exchangers and minimized piping.

Major Trade-offs:

- Performance vs. Weight: Larger heat exchangers can be more effective but heavier.
- Complexity vs. Reliability: Parallel loops enable better control but add complexity.
- Reliability vs. Redundancy: Duplicated hardware increases weight, but improves safety.

4a. Logical Elements to Physical Components Mapping

See Table AT.52

4b. Key Design Constraints

- FC Stack, DC/DC, and Air Conditioning HX inlets require TMS coolant <70°C.
- Hydrogen Conditioning HX requires TMS inlet >60°C.
- Ram Air HX capacity must handle up to -433 kW of heat rejection at cruise.
- Total system pressure drop $\leqslant \sim 3.5$ bar pump boost (typical) and piping <0.15 bar total.

4c. Architecture Configuration Analysis: We evaluate three possible configurations:

1. *Serial Configuration:* All heat exchangers in one continuous loop
2. *Parallel Configuration:* Each major load has an independent branch fed by a common manifold
3. *Hybrid Configuration:* Some loads in parallel branches, merging back through a common final heat exchanger or manifold

Decision Criteria Applied:

- Thermal Performance: Ensure inlet <70°C for cooling HX, ~60-70°C for H_2 conditioning
- Pressure Drop: Keep total Δ P manageable by the pump (2-3.5 bar range), no single HX >0.4 bar
- Flow Control Complexity: Balanced approach for partial load management/fault tolerance
- Redundancy Requirements: Potential for a second pump for critical FC stack cooling
- Integration Constraints: Aircraft packaging, minimal piping runs, minimal weight

Table AT.52.: TMS generation report - logical elements to physical components mapping

Logical El. ID	Logical El. Name	Physical Component	Component Specifications	Selection Rationale	Design Constraints
LE-001	HX	Plate-Fin HX (FC Stack)	Custom geometry, aircraft-grade aluminum, 0.2-0.3 bar drop	Very high effectiveness, compact design for aircraft space	Space envelope near FC stack, $< 70°C$ inlet requirement
LE-001	HX	Plate-Fin HX (DC/DC)	Scaled-down geometry, 0.1-0.2 bar drop	Similar design approach, smaller capacity & footprint	25 kW load requirement, inlet $< 70°C$
LE-001	HX	Plate-Fin HX (Air Cond)	Custom geometry for 36.3 kW capacity, 0.2 bar drop	Operates at high external temps (up to 129°C), requires robust core	Inlet $< 70°C$ for TMS side
LE-001	HX	Plate-Fin HX (H2 Cond)	High temperature gradient handling, 0.2 bar drop	For hydrogen warming from -252°C up to 60°C, needs careful design	TMS inlet $\geq 60°C$, structural safety for cryogenic interfacing
LE-001	HX	Plate-Fin HX (Ram Air)	High capacity for ∼433 kW rejection, 0.3-0.4 bar drop	Main system heat sink, must handle large load quickly	Installed in aircraft exterior duct, harsh environment exposure
LE-002	Coolant Pump	Electric Centrifugal Pump	Variable speed (50-200 LPM), 2.5-3.5 bar boost	Precise flow control, lightweight aerospace motor	Power consumption ∼4-7 kW, 75% efficiency assumed
LE-003	Control Valve	Solenoid/ Flow Control Valve	0.05-0.1 bar drop fully open, fast actuation	Referenced design from commercial aerospace applicat.	Temperature endurance up to 100°C, minimal leak
LE-005	Thermal Control Unit	Digital TMS Controller	MPC/SMC fault-tolerant control loops, multiple sensor inputs	From literature on advanced TMS control in aircraft environment	Interfaces: ARINC-429 or CAN bus, real-time OS
LE-007	Load Shedd. Funct.	Integrated TMS-FCU Logic	Software-based load priority algorithms	Monitors TMS capacity, aircraft power demands	Must coordinate with flight controls, pilot overrides
LE-008	Alerting System	Flight Deck Alert Integration	Central alert prioritization, standard cockpit EICAS interface	Follows standard flight deck design guidelines	Minimal nuisance alerts, must be consistent with rest of cockpit
LE-005	Thermal Control Unit	Expansion/ Deion. Tank & Filter Module	∼5-10 L capacity, integrated deion resin, ∼0.1-0.2 bar drop	Specifically designed for water-glycol mixture & aircraft altitude changes	Maintains low conductivity, handles volumetric expansion with altitude

Selected Configuration - Hybrid Approach:
The chosen hybrid configuration provides optimal balance between thermal performance and system complexity. The pump feeds a filter/deionization module before splitting into two main branches:

- Branch A: FC Stack HX (440 kW) in series with DC/DC HX (25 kW)
- Branch B: Air Conditioning HX (36.3 kW) in series with H_2 Conditioning HX (-68.1 kW)

Both branches converge before the Ram Air HX for final heat rejection, followed by the expansion/deionization tank before returning to the pump.

Configuration Rationale: By grouping the FC Stack and DC/DC (both requiring <70°C inlet) on Branch A, the system ensures they receive the coolest flow. The Air Conditioning and H_2 Conditioning are on Branch B, benefiting from near 60-70°C coolant. This approach balances flows, ensures the H_2 conditioner's >60°C requirement, and consolidates final heat dumping in a single Ram Air HX.

5. Design Evolution Through Iterative Analysis
Demonstration table (Table AT.53) for the Hybrid Configuration. Detailed steps are shown inline. (Temperatures in °C, Pressures in bar, Mass Flow in kg/s).

Table AT.53.: TMS generation report - thermodynamic analysis (hybrid configuration)

St.	Comp.	Configuration	P_{in} (bar)	T_{in} (°C)	P_{Out} (bar)	T_{Out} (°C)	Δ P (bar)	Mass Flow (kg/s)	Design Reasoning
1	Main Pump	Main line	1.0	50	3.5	52	+2.5	22.0	Pressurize system to overcome HX/-valve/piping drops
2	Filter & Deionis.	Main line	3.5	52	3.3	52	-0.2	22.0	Coolant cleaning & deion, keep conduct. <3μS/cm
3	Manifold	Split into Branch A & B	3.3	52	3.25	52	-0.05	22.0 total	Flow splitting to separate branches
4a	FC Stack HX	Branch A	3.25	52	2.95	62	-0.3	12.6	Primary load removed here
4b	DC/DC HX	Serial after FC	2.95	62	2.85	65	-0.1	12.6	Supports power electronics cooling
5a	Air Cond HX	Branch B	3.25	52	3.05	60	-0.2	5.5	Cabin air cooling demand
5b	H_2 Cond HX	Serial after Air Cond.	3.05	60	2.90	55	-0.15	5.5	Warms hydrogen from cryogenic temps
6	Merge Branches	Common line to Ram Air HX	2.85 /2.90	≈65 /55	2.70	≈60	-0.1	≈18.1 total (A+B)	Flow merges into final cooling stage
7	Ram Air HX	Final cooling	2.70	60	2.40	50	-0.3	18.1	Overall heat rejection to atmosphere
8	Expansion/ Deion Tank	Return line	2.40	50	2.35	50	-0.05	18.1	Manages volume changes, ensures low conductivity
9	Return to Pump	Main line	2.35	50	1.0	50	-1.35	18.1	System loop completes

We validate each component's pressure drop using the provided JavaScript function `validateRealisticPressureDrops()`. The summary table (see Table AT.54) below confirms compliance with typical aerospace TMS design limits:

Table AT.54.: TMS generation report - pressure drop validation analysis

Component	**Calculated Δ P (bar)**	**Accept. Range (bar)**	**Status**	**Comments**
Ram Air HX	0.30	0.2-0.4	✓	Within recommended max 0.4 bar
FC Stack HX	0.30	0.1-0.3	✓	Largest load, borderline upper limit
DC/DC HX	0.10	0.1-0.3	✓	Minimal load, easy to cool
Air Cond HX	0.20	0.1-0.3	✓	Moderate load
H_2 Cond HX	0.15	0.1-0.3	✓	Warming flow is manageable
All Valves	0.05-0.08	0.05-0.12	✓	Flow control and isolation valves tested
Piping Total	**0.10**	**0.05-0.15**	✓	**Within limit for entire circuit**
Filter/Deion	0.20	0.1-0.2	✓	Acceptable for this flow rate

During the design process, three principal issues were noticed:

- Potential Overheating in FC Stack HX under peak loads : Resolved by slightly increasing mass flow from 12.6 to $\sim$ 13.0 kg/s, verified by re-running the mass flow calculations.
- Pressure Drop near 0.4 bar in Ram Air HX : We resized the Ram Air HX geometry to reduce it to $\sim$ 0.3 bar at the design flow.
- Need to ensure the H_2 Cond HX inlet is >60°C : Confirmed via partial parallel design so that coolant from the pump and filter is not drastically cooled before reaching the H_2 line.

Validated System Design

The final coolant flow path of the Single Hybrid Loop is: Pump → Filter/Deion → Main Isolation Valve → Manifold → (Branch A: FC Stack HX → DC/DC HX) & (Branch B: Air Cond HX → H2 Cond HX) → Merge → Ram Air HX → Expansion/Deion Tank → Return

Table AT.55.: TMS generation report - final validated system design specification

Stg.	Component Name	Function	Mass Flow (kg/s)	P (bar)	T (°C)	Valve Type & Funct.	Design Notes
1	Main Pump	Pressurize system	∼22.0	1.0 → 3.5	50 → 52	N/A	Primary flow source, ∼75% eff.
2	Filter/Deion Module	Coolant cleaning	22.0	3.5 → 3.3	52	N/A	Deion & filtration, ∼0.2 bar drop
3	Main Isolation Valve	System shutdown	22.0	3.3 → 3.2	52	Isolation-Fail-closed	Emergency shutoff capability
4	Flow Splitter	Split to branches	22.0 → (A:12.6 + B:5.5)	3.2	52	N/A	Equal pressure distribution
5a	FC Branch Flow Valve	Regulate Branch A flow	12.6	3.2 → 3.1	52	Fail-open	Precise flow for FC stack
5b	FC Stack HX	Heat pickup (440 kW)	12.6	3.1 → 2.8	52 → 62	N/A	Primary load removal
5c	FC Branch Isolation Valve	Branch A exit	12.6	2.8 → 2.7	62	Fail-closed	Isolate for maintenance if needed
5d	DC/DC HX	Heat pickup (25 kW)	12.6	2.7 → 2.6	62 → 65	N/A	Cooling for power electronics
5e	Branch A Merge Valve	Prevent backflow	12.6	2.6 → 2.55	65	Fail-closed	Check valve to unify flow
6a	Air Cond. Branch Flow Valve	Regulate Branch B flow	5.5	3.2 → 3.1	52	Fail-open	Cabin AC flow control
6b	Air Cond. HX	Heat pickup (36.3 kW)	5.5	3.1 → 2.9	52 → 60	N/A	Cabin climate control
6c	H_2 Cond. HX	Heating load (-68.1 kW)	5.5	2.9 → 2.75	60 → 55	N/A	Warms cryogenic hydrogen
6d	Branch B Merge Valve	Prevent backflow	5.5	2.75 → 2.70	55	Fail-closed	Check valve to unify flow
7	Ram Air HX	Final heat rejection	∼18.1	2.70 → 2.40	∼60 → 50	N/A	Rejects up to ∼433 kW to ambient
8	Expansion Tank	Volume compensation., conductivity control	18.1	2.40 → 2.35	50	N/A	Adds ≈0.05 bar drop, ensures fluid quality