UNIVERSITY OF TWENTE.

Trustworthiness by Design: Structuring Trustworthy RAG-Augmented LLM Assistants via Enterprise Architecture

by

Anton S. Tsankov

A thesis submitted to the
Faculty of Electrical Engineering, Mathematics and Computer Science (EEMCS)
in partial fulfilment of the requirements for the degree of

MSc in Business Information Technology

Faculty of Electrical Engineering, Mathematics and Computer Science (EEMCS)

University of Twente

Enschede, Overijssel, The Netherlands

July 2025

ABSTRACT

The integration of Explainable Artificial Intelligence (XAI) into proactive chatbot systems powered by Retrieval Augmented Generation (RAG) (a enhances language models by retrieving relevant external documents to generate more informed and accurate responses) has the potential to transform educational environments by improving transparency, trustworthiness, and usability. This study reviews the architectural components, design frameworks, and evaluation methods required to embed trustworthiness into chatbot systems. Through a systematic exploration of XAI frameworks and mechanisms like Retrieval-Augmented Generation Assessment Suite (RAGAS) (evaluates RAG pipelines by scoring their outputs based on relevance, faithfulness, and answer quality) and Retrieval-Augmented Generation Evaluation (RAGE) (a scoring system that evaluates trustworthiness in RAG systems by measuring input and output faithfulness, and source attribution), the following research identifies key patterns and trade-offs involved in designing scalable, compliant, and user-personalized solutions RAG-powered Large Language Model (LLM) assistants. In addition, the study tests different Enterprise Architecture (EA) frameworks and system architectures to assess their influence on the performance and integration of Trustworthy Artificial Intelligence (TAI) in chatbot systems. Thus, this study's main research questions focus on analyzing the impacts of different EA approaches on the integration of TAI, scoped to RAG and LLMs, in Artificial Intelligence (AI)-powered assistants. It explores how TAI can enhance the interpretability of RAG-based outputs, identifying the most effective methods for evaluating chatbot transparency, and examining the role of TAI in fostering trustworthiness and adoption within educational systems. Additionally,the following article investigates the challenges of aligning XAI models with Information Technology (IT) management frameworks and governance standards, with a particular focus on compliance.

AUTHOR'S DECLARATION

I hereby declare that this thesis consists of original work of which I have authored. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I authorize the University of Twente to lend this thesis to other institutions or individuals for the purpose of scholarly research. I further authorize University of Twente to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research. I understand that my thesis will be made electronically available to the public.

Anton S. Tsankov

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to the following individuals for their invaluable support throughout the course of this project: Firstly, I want to express my most sincere appreciation to my thesis supervisors, Marcos Machado, Wallace Ugulino, Patricia Rogetzer and Gayane Sedrakyan, for sharing their expertise and guiding me in the development of my thesis. In particular, I would like to thank Marcos and Wallace for introducing me to the fields of my project, providing timely, comprehensive and consistent feedback, and for the moral encouragement given during the past year.

Thank you all for your contribution to this thesis and for guiding me toward the accomplishment of this significant milestone in my life. I hope you enjoy the reading!

Disclaimer: Portions of this work, including writing (QuillBot[1]), idea development (ChatGPT[2]), and experimentation support (Google Gemini[3]), have involved the use of generative AI tools as part of the research process.

[1] QuillBot is an AI-powered writing and paraphrasing assistant available at https://quillbot.com
[2] ChatGPT is a conversational AI developed by OpenAI, accessible at https://chat.openai.com
[3] Google Gemini is a generative AI platform by Google, available at https://gemini.google.com

CONTENTS

List of Figures

List of Tables

ABBREVIATIONS

ADM Architecture Development Method.

AI Artificial Intelligence.

ANN Approximate Nearest Neighbor.

BIM Binary Independence Model.

BITAI Business-IT Alignment Index.

BIZBOK Business Architecture Body of Knowledge.

CRISP-ML Cross-Industry Standard Process for Machine Learning.

DES Discrete-Event Simulation.

DoDAF Department of Defense Architecture Framework.

DSR Design Science Research.

EA Enterprise Architecture.

ESB Enterprise Service Bus.

EU European Union.

FEAF Federal Enterprise Architecture Framework.

GDPR General Data Protection Regulation.

GSC Goal Satisfaction Index.

HNSW Hierarchical Navigable Small World.

IAF Integrated Architecture Framework.

IDF Inverse Document Frequency.

IF Input Faithfulness.

IT Information Technology.

KQ Knowledge Question.

LCR Layer Coverage Ratio.

LLaMA Large Language Model Meta AI.

LLM Large Language Model.

LMS Learning Management System.

MAE Mean Absolute Error.

MDE Model-Driven Engineering.

ML Machine Learning.

MODES-EA Model-Driven Framework for Discrete-Event Simulation of Enterprise Architecture Models.

MSE Mean Squared Error.

MVC Model View Controller.

NMSE Normalized Mean Squared Error.

OF Output Faithfulness.

PDF Portable Document Format.

R2 R^2 Coefficient of Determination.

RAG Retrieval Augmented Generation.

RAGAS Retrieval-Augmented Generation Assessment Suite.

RAGE Retrieval-Augmented Generation Evaluation.

RMSE Root Mean Squared Error.

RQ Research Question.

SA Source Attribution.

SAMM Strategic Alignment Maturity Model.

SLR Systematic Literature Review.

SOA Service Oriented Architecture.

TAI Trustworthy Artificial Intelligence.

TAM Technology Acceptance Model.

TF Term Frequency.

TheilU Theil's U Statistic.

TOGAF The Open Group Architecture Framework.

TR Traceability Ration.

VVA Verification, Validation, and Accreditation.

XAI Explainable Artificial Intelligence.

ZF Zachman Framework.

1

INTRODUCTION

1.1. RESEARCH BACKGROUND

With increasing use of AI systems and technologies within the educational domain , the ability of XAI to provide explanations of AI decisions and outputs has garnered significant attention [2]. XAI aims to make Machine Learning (ML) models understandable and interpretable for human users, thereby improving trust and fostering more effective human-AI collaboration [3].

An essential aspect in this context is the development of TAI systems, which are characterized by their reliability, ethical alignment, transparency, fairness, and respect for user privacy [4, 5]. TAI systems deliver outputs accompanied by transparent explanations, thus enabling informed and confident user engagement with AI technologies [6].

AI-driven educational tools, especially chatbots using LLMs, require XAI to ensure transparency and trust. This research explores integrating XAI into RAG chatbots, assessing how EA impacts their explainability and user trust in education.

Proactive chatbots, powered by LLMs, have emerged as valuable tools in educational systems. These tools are designed to provide assistance to learners via dynamic dialogues [7]. However, one major challenge is ensuring that these systems are understandable to their users (including students, teachers, educational supporting staff, and IT managers) [8]. Previous studies have shown that the lack of transparency in AI-powered systems can reduce user trust, impair decision-making, and hinder the widespread adoption of such technologies. [3, 9].

1.2. RESEARCH MOTIVATIONS AND OBJECTIVES

RAG has become a prominent approach to enhance the capabilities of chatbots by augmenting their responses with external knowledge [10]. RAG is a technique that combines two key components: a retriever and a generator. The retriever selects relevant information from a

large knowledge base, which is then provided to the generator, enabling it to produce contextually enriched and more accurate responses. However, the integration of explainable models within RAG-powered systems remains a challenge, especially when it comes to explaining the rationale behind generated responses [6, 11]. Understanding how to make such systems interpretable to all stakeholders (students, teachers, educational supporting staff, and IT managers) is crucial for ensuring their successful deployment in educational settings.

Therefore, the following research aims to compare various EA approaches, including the modeling of different architectural patterns, integration structures, and methodologies, for integrating XAI within AI-powered educational assistants. The study will also investigate the impact of these approaches on XAI, focusing transparency, trust, scalability, and IT governance compliance in particular.

To achieve these goals, the following Research Question (RQ)s are created.

- **RQ1:** *How can TAI be integrated into RAG-powered LLM educational assistants to enhance transparency, and trust?*
- **RQ2:** *How do different Enterprise Architecture approaches and patterns influence the integration and output of TAI in AI-powered educational assistants?*

The relevance of this research question derives from the gap identified within the covered literature, see Chapter 2. Specifically, while the literature highlights the increasing relevance of TAI and its application in educational systems, it reveals a lack of structured methodologies that explore how different enterprise architectural patterns influence the integration of TAI within AI-powered educational environments.

1.3. METHODS

This study employs a combination of Design Science Research (DSR) and the Cross-Industry Standard Process for Machine Learning (CRISP-ML) to systematically investigate the integration of XAI within educational assistants. DSR provides a rigorous framework for artifact creation, evaluation, and iterative refinement, ensuring that practical and theoretical contributions are tightly interwoven. CRISP-ML guides the machine learning life-cycle, from data collection and pre-processing to model training, evaluation, and quality assurance, enabling reproducible and robust experimentation.

Analytical methods include quantitative and qualitative assessment of model output faithfulness, citation accuracy, and trust metrics derived from controlled experiments with RAG architectures. Experimental setups are designed to compare the effects of model scale, retrieval strategies (lexical, semantic, hybrid), and architectural patterns on explainability and transparency. Simulation tools are used to evaluate enterprise architecture frameworks and software design patterns, focusing on scalability, traceability, and governance compliance in RAG-powered LLM educational systems.

These combined methods allow the study to assess both the technical and organizational dimensions of explainability, bridging empirical model evaluation with system-level architectural analysis.

1.4. OUTLINE

The document is structured to provide a comprehensive and logical progression through the research objectives and findings. It begins with Chapter 1, which presents the research background, motivation and objectives, methodology overview, and a brief outline of the document.

Next, Chapter 2 (Literature Review) details the methodology for reviewing existing research, deconstructs the relevant academic landscape, identifies predominant themes, explores managerial implications, and concludes with a thematic summary of key findings.

Chapter 3 (Methodology) then elaborates on the chosen research frameworks and analytical approaches.

Subsequent chapters focus on two main experimental domains: TAI, which covers data collection, environmental setup, data processing, and RAG experimental configurations; and EA, which addresses data structure, artefact design, framework specification, proposed architectures, and experimental setups.

The Aligning Experimental Approaches chapter synthesizes the individual contributions from both domains, recaps experimental objectives, and explores integrated insights, positioning trust as a core outcome and design consideration.

The Results (Chapter 7) present findings from both AI and architectural experiments, followed by the Discussion (see Chapter 8, which directly addresses the research questions.

The document concludes with the Conclusion chapter summarizing lessons learned, contributions, and limitations, and the Future Work chapter proposing next steps toward enhancing trust, explainability, and architectural transparency in AI-powered educational systems.

2

LITERATURE REVIEW

The goal of the following Systematic Literature Review (SLR) is to systematically explore the components and architectural patterns necessary to integrate XAI into RAG-powered LLM proactive chatbot systems within educational environments. In addition, we aim to understand how XAI can be applied to improve the transparency, interpretability, and trustworthiness of RAG-powered chatbots in educational setting [1] .

Additionaly the SLR aims to explore various EA approaches, including the modeling of different architectural patterns, integration structures, and methodologies, for integrating XAI within RAG-powered LLM chatbot assistants. Furthermore it will investigate the impact of these approaches on transparency, trust, scalability, and IT governance compliance across diverse stakeholders, including students, educators, and IT managers.

To achieve these goals, the SLR aims to answer the following Knowledge Question (KQ)s:

1. **KQ1:** What are the key components and architectural patterns required to integrate XAI into proactive chatbot systems in educational environments?
2. **KQ2:** How can LLMs with RAG be made more explainable for different stakeholders in educational contexts?
3. **KQ3:** What are the challenges in aligning XAI models with IT management frameworks for educational system governance and compliance?
4. **KQ4:** How can EA frameworks support and influence the integration of XAI into educational systems?

Answering the set of KQs contributes to the design of AI systems that ensure transparency without compromising the efficiency of XAI education-supported systems. Addressing the chal-

[1] While this thesis reviews XAI methods in the literature, it adopts a broader perspective by focusing on TAI, which encompasses not only explainability but also aspects such as reliability, fairness, and transparency.

lenges [2] that come with such systems support the creation of explainability strategies that meet the needs of various stakeholders, making XAI explanations accessible and meaningful to students, teachers, and IT managers. Thus, the SLR below presents insights into how governance frameworks can incorporate XAI models, while helping IT managers assess the risks and benefits of scaling XAI in educational settings, supporting informed decision-making.

2.1. METHODOLOGY

Within the SLR we follow a structured and methodical approach to ensure adequate coverage of our analysis. Our approach is designed to explore the intersection of XAI, RAG, and their applications in educational environments, focusing on proactive support systems and LLMs. The methodology consists of several key steps to identify, select, and synthesize relevant research articles.

The initial step is to define the research problem and establish both clear research and knowledge questions based on the core objectives of the study. The next step involves conducting a search of the literature using the search queries to capture relevant studies (see Tables A4, and A5).

Once the search is conducted, we apply inclusion and exclusion criteria, as seen in Tables 2.1 and 2.2, to filter irrelevant or low quality studies. While mostly the same, the two criteria sets, differ in temporal restrictions, due lack of literature post 2021 for EA. These criteria ensure that only high-quality, peer-reviewed research is included in the review. Following the filtering phase, the quality of each selected study is evaluated using a systematic assessment framework that includes factors such as research design, methodology, and relevance to research questions. The next phase involves data extraction, where key findings, including system architectures, XAI techniques, evaluation metrics, and educational impacts, are collected and categorized, see Tables A1.

The final steps involve data analysis and synthesis. In this phase, we integrate the findings from the selected studies, identifying key trends, gaps in research, and insights related to the application of XAI and RAG in educational systems. This allows us to draw comprehensive conclusions about the impact of these technologies on learning outcomes, trust, transparency, and system scalability. A thematic analysis is conducted to categorize major trends in the literature, ensuring a structured synthesis of findings.

For our search, we primarily use the Scopus [2] database , which provides an extensive repository of academic articles and advanced search features. The advanced search functionality is employed to query the database using the keywords defined earlier. To further validate our dataset, we complement our search with Google Scholar [3] to capture potentially overlooked

[2] Scopus as in the abstract and citation database that covers a wide range of academic disciplines, including science, technology, medicine, social sciences, and arts and humanities.

[3] Google Scholar as in the freely accessible search engine that indexes scholarly literature, including articles, theses, books, and conference papers, across various academic disciplines.

Table 2.1: XAI - Summary of article selection criteria

Criteria	Decision
Inclusion of pre-defined keywords in title, abstract, or keyword list	Inclusion
Article publication in a scientific journal or conference proceedings	Inclusion
Article written in English	Inclusion
Article published before 2021	Exclusion
Duplicates of an original article	Exclusion
Relevance of abstract, title, and content to research objective	Exclusion
Unavailability of the article online for free	Exclusion

Table 2.2: EA - Summary of article selection criteria

Criteria	Decision
Inclusion of pre-defined keywords in title, abstract, or keyword list	Inclusion
Article publication in a scientific journal or conference proceedings	Inclusion
Article written in English	Inclusion
Article published before 2019	Exclusion
Duplicates of an original article	Exclusion
Relevance of abstract, title, and content to research objective	Exclusion
Unavailability of the article online for free	Exclusion

studies. These studies are then filtered and categorized to ensure that only the most pertinent research is considered in the final analysis, as seen in Figure 2.1.

2.2. DECONSTRUCTING THE RESEARCH LANDSCAPE

To facilitate comprehension whist granting a complete perspective, all literature consulted for the following study is indexed in Table A1. The table provides an overview of each paper, highlighting the settings in which the respective model was employed; the main objectives of the research, the data-driven techniques applied;the evaluation methods utilized to gauge the efficiency of the approach

2.2.1. TEMPORAL DISTRIBUTION OF LITERATURE

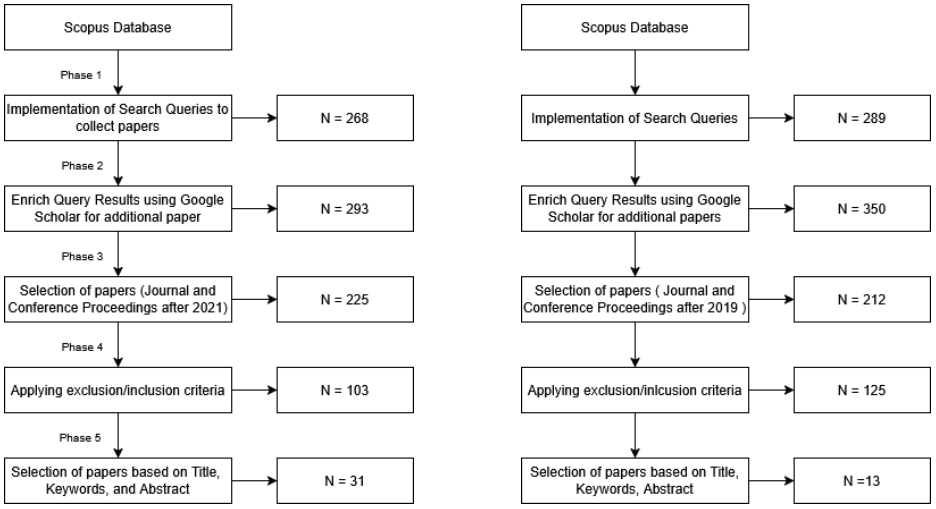
We begin by performing an examination on the temporal progression of the scholarly discourse on XAI and LLM integration in educational settings. Figure 2.2 illustrates a pattern, showcasing a rise in the number of papers published in recent years.

Although low in 2023, the continued relevance of the topic is evident, particularly when considering the absence of any publications on this subject prior 2021 to the best of our knowledge. Therefore, observing an increase in publications in 2022 and 2023 suggests a possible rise in interest and use of XAI, in particular LLMs, in educational institutions.

2.2.2. JOURNAL DISTRIBUTION OF LITERATURE

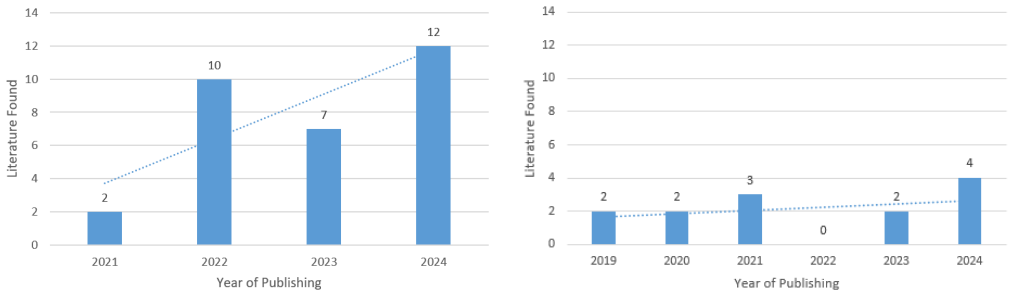
The next analysis focuses onto the journal distribution of collected literature. Understanding said dispersion provides insights into the diversity and reach of the research community addressing these topics.

Table 2.3 shows an overview of XAI-related articles, outlining properties of the selected liter-



(a) XAI - Illustration of the article selection process (b) EA - Illustration of the article selection process

Figure 2.1: Selection process for XAI and EA articles and queries



(a) XAI - Temporal distribution of research papers (b) EA - Temporal distribution of research papers

Figure 2.2: Temporal distribution for XAI and EA articles and queries

ature (respective journals, citation counts, impact factors). The article by Wang et al. [12] stands out with the highest citation count of 2530, outlining a large influence in the field of educational technology. Articles by Holmes et al. [5] and Khosravi et al. [2], are published in the *International Journal of Artificial Intelligence in Education* and *Computers and Education: Artificial Intelligence*. Part of the recorded studies have appeared in a variety of journals and open databases, such as ArXiv [4] preprints [13] and the *Proceedings of the 27th International Conference on Intelligent User Interfaces* [14], suggesting that researchers are diversifying their publication outlets rather than adhering to a specific journal.

Table 2.3: Journal distribution of XAI-related articles

Row Nr	Study	Journal	# Citations	Impact Factor
1	Homes (2022)[5]	International Journal of Artificial Intelligence in Education	–	-
2	Hooshyar (2024)[15]	IEEE Access	–	-
3	Fiok (2022)[11]	The Journal of Defense Modeling and Simulation	133	2.23

[4] ArXiv as in the free, open-access repository that hosts preprints of research papers in fields like physics, mathematics, computer science, and more. It allows researchers to share their findings before formal peer review.

4	Swammy (2022)[13]	arXiv preprint arXiv:2207.00551	4	–
5	Khosravi (2022)[2]	Computers and Education: Artificial Intelligence	–	–
6	Yang (2021) [16]	Computers and Education: Artificial Intelligence	–	–
6	Wang (2024)[12]	British Journal of Educational Technology	2530	6.91
7	Ogata (2024)[17]	Research and Practice in Technology Enhanced Learning	–	–
8	Kamawar (2024)[18]	Proceedings of the 12th International Conference on Human-Agent Interaction	1	–
9	Tiddy (2022)[19]	Artificial Intelligence	–	–
10	Chou (2022) [20]	Information Fusion	59	3.524
11	Cambria (2023) [21]	Information Processing & Management	103111	3.015
12	Zerilli et al. (2022)[3]	Patterns	4	3.85
13	Gajos et al. (2022)[14]	Proceedings of the 27th International Conference on Intelligent User Interfaces	–	–
14	Papenmeier et al. (2022) [22]	ACM Transactions on Computer-Human Interaction (TOCHI)	33	3.346
15	Saarela (2024) [8]	Applied Sciences	8884	2.922
16	Kostopoulos et al. (2024) [23]	Electronics	2842	2.207
17	Alonso et al. (2019) [24]	International Workshop on Higher Education Learning Methodologies and Technologies Online	–	–
18	Duan et al. (2024) [6]	Education and Information Technologies	3075	2.944
19	Liu et al. (2024) [25]	Trust and Inclusion in XAI-Mediated Education: Where Human Learning Meets Learning Machines	–	–
20	Casalino et al. (2023) [26]	International Conference on Higher Education Learning Methodologies and Technologies Online	–	–
21	Hennekeuser (2024) [10]	International Journal of Artificial Intelligence in Education	–	–
22	Vera et al. (2024) [7]	International Conference on Human-Computer Interaction	–	–
23	Kasneci et al. (2023) [27]	Learning and Individual Differences	–	–
24	Renz et al. (2021) [28]	Technology Innovation Management Review	–	–
25	Bhat et al. (2024) [29]	Proceedings of the 2024 ACM Designing Interactive Systems Conference	–	–
26	Chaushi et al. (2023) [30]	World Conference on Explainable Artificial Intelligence	–	–
27	Jafarzade et al. (2023) [31]	2023 5th International Conference on Problems of Cybernetics and Informatics (PCI)	–	–
28	Uglev et al. (2023) [32]	International Conference on Intelligent Tutoring Systems	–	–
29	Kaur et al. (2022) [4]	ACM Computing Surveys (CSUR)	–	–

Table 2.4 provides an overview of recent literature at the intersection of EA and XAI, highlighting venues of publication, citation counts, and where available, impact metrics. Among the listed studies,[33] stand out with 45 citations, suggesting early recognition within applied domains. [34] and [35] also show moderate engagement with 22 and 27 citations respectively, indicating steady, but limited academic interest in their contributions.

Several papers are published in conference proceedings rather than traditional journals, such as PICMET [36], EDOCW [34], and IC3INA [37], underlining the field's orientation toward practice-driven insights and emerging technologies. A notable portion of the literature is hosted in domain-specific journals or thematic books like Sayles et al.[38]. This showcases a cross-disciplinary and evolving publishing landscape. The relatively low citation numbers across most entries may point to the nascent and exploratory nature of research at its intersection, with EA scholars beginning to address with the integration of XAI into complex organizational systems.

Table 2.4: EA Studies Related to XAI Integration

Row Nr	Study	Journal/Conference	# Citations	Impact Factor
1	Kamalabai et al. (2024) [36]	2024 Portland International Conference on Management of Engineering and Technology (PICMET)	1	–
2	Sayles (2024) [38]	*Principles of XAI Governance and Model Risk Management*	–	–
3	Rittelmeyer et al. (2021) [34]	2021 IEEE 25th International Enterprise Distributed Object Computing Workshop (EDOCW)	22	–
4	Ilin et al. (2019) [33]	*Energy Management of Municipal Transportation Facilities and Transport*	45	–
5	Dilnutt (2023) [39]	*Enterprise Architecture Professional Journal*	–	–
6	Zimmermann et al. (2020) [40]	*Research Challenges in Information Science: RCIS 2020*	5	–
7	Malott (2024) [35]	27	–	–
8	Fitriani et al. (2023) [37]	2023 International Conference on Computer, Control, Informatics and its Applications (IC3INA)	0	–
9	Denni et al. (2024) [41]	*International Conference on Business and Technology*	7	–
10	Wang et al. (2020) [42]	–	3	–
11	Sandkuhl (2021) [43]	*Enterprise Engineering Working Conference*	7	–
12	Martel et al. (2021) [44]	*INFORMATIK 2020*	7	–
13	Takeuchi et al. (2019) [45]	*Procedia Computer Science*	14	–

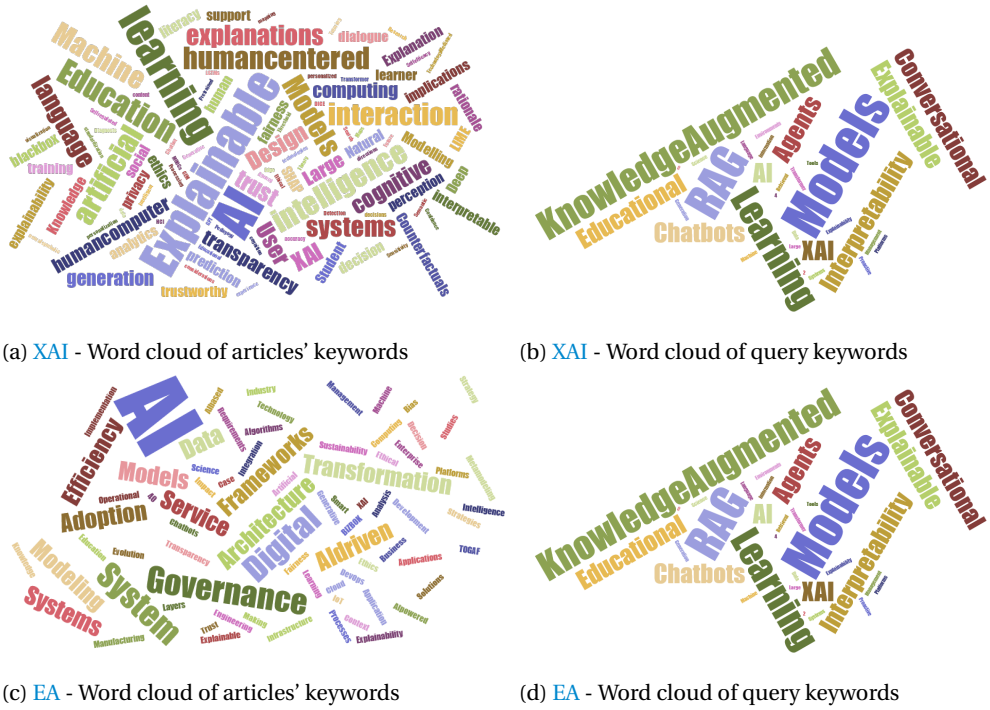
The broad and uneven distribution of literature across a range of journals and conferences, along with fluctuating citation counts and generally limited reporting on impact factors, reflects the interdisciplinary and still-maturing nature of these research areas. Such variability suggests both a wide applicability and an evolving interest hinting at the fields' expanding significance.

2.2.3. DISTRIBUTION OF PUBLICATIONS BASED ON KEYWORDS

The analysis of the studied papers presents "Explainable XAI" as the predominant keyword, followed by "Learning," "Human-centered," "Interaction," and "Artificial Intelligence." These keywords suggest a concentrated interest in XAI systems that prioritize transparency, user comprehension, and human-centric design. Additionally, other frequently occurring terms such as "Education," "Systems," "Explanations," "Transparency," and "Cognitive" indicate a strong focus on the integration of XAI into educational frameworks, emphasizing trustworthiness, knowledge dissemination, and the role of XAI in shaping learning experiences.

Figure 2.3 presents clouds showcasing the most prevalent keywords in the analyzed literature. The word size in the cloud is indicative of the relative frequency of each keyword.

As observed in Figures 2.3a and 2.3b , "Explainable XAI" emerges as the most commonly used term, followed closely by "Learning," "Human-centered," "Interaction," and "Artificial Intelligence." These findings suggest that research in the following area is highly focused on designing XAI-driven educational system. The frequent recurrence of "Education," "Systems," "Explana-



(a) XAI - Word cloud of articles' keywords

(b) XAI - Word cloud of query keywords

(c) EA - Word cloud of articles' keywords

(d) EA - Word cloud of query keywords

Figure 2.3: Word clouds for XAI and EA articles and queries

tions," "Transparency," and "Cognitive" highlights potential emphasis on making XAI systems more understandable.The prominence of "Learning" and "Human-centered" approaches suggests a movement towards XAI models that support student learning by providing interpretable insights, and transparent decision-making processes. The inclusion of "Interaction" implies that XAI-driven educational tools are increasingly expected to engage users dynamically, rather than merely presenting static information.The prevalent presence of "Transparency" and "Explanations" presents a growing need for XAI models that can justify their recommendations, predictions, or assessments. This aligns with ongoing efforts to ensure that XAI-powered learning platforms and tutoring systems are not perceived as "black boxes", but rather as tools that can articulate their reasoning in a way that can be trusted.

As seen in Figures 2.3c and 2.3d "XAI" stands out as the most frequently used term, followed by "Governance," "System," "Digital," "Transformation,", "Framework,". This suggests that the literature focuses heavily on XAI-driven approaches within EA, particularly in areas like digital transformation, system integration, and data governance. Terms such as "Frameworks," "Architecture," and "Adoption" further point to structured approaches for implementing XAI in enterprise settings. Furthermore, largely prominent terms such as "Models", "AIDriven," "KnowledgeAugmented," and "Explainable" indicate a focus on transparency, interpretability, and explainability in knowledge-enhanced XAI systems.

2.3. PREDOMINANT THEMES IN LITERATURE

The integration of XAI into educational and learning environments has gained significant attention due to its potential to enhance transparency, trust, and learning outcomes [2, 14]. Given the increasing reliance on XAI-driven educational tools, it is essential to critically analyze these themes from multiple perspectives to ensure robust, scalable, and interpretable XAI solutions [27]. To further illustrate the predominant themes in literature, in addition to overview tables (see Tables 2.5 and A1 for XAI; Tables 2.6 and A2 for EA).

Literature on XAI in educational setting highlights key areas influencing system adoption and user trust. Transparent XAI-powered educational assistants improve usability and reduce cognitive load when explanations are structured and tailored to user expertise [3, 10, 12, 14, 22]. XAI systems enhance student engagement and self-efficacy in problem-solving when designed for interactive learning [6, 7, 27]. However, post-hoc explainability methods like SHAP and LIME show limited interpretability and consistency, whereas counterfactual techniques (e.g., DiCE, CEM) offer higher fidelity with greater computational cost [13, 15]. Ethical and legal compliance—particularly in relation to General Data Protection Regulation (GDPR)—requires governance structures to guide responsible XAI deployment [4, 5, 23]. Additional studies emphasize algorithmic literacy [2, 21, 29], fairness [8, 11, 20], and the importance of self-explanation mechanisms for knowledge retention [16, 17]. Scalability across institutions remains a challenge, often requiring trade-offs between transparency and computational efficiency [26, 31].

XAI adoption influences EA layers including business processes and technology infrastructure [34]. EA frameworks support sustainable and ethical XAI deployment through aligned governance models [36, 38]. EA also facilitates digital transformation and supports the integration of XAI in smart manufacturing and enterprise analytics [33, 39, 42]. The evolution of EA to include cloud computing, IoT, and Generative XAI is seen as necessary for future scalability and flexibility [40, 41]. Frameworks like TOGAF assist in structuring XAI adoption and aligning it with enterprise capabilities [37]. EA further aids in XAI requirements engineering and scalable system development through best practices in DevOps and data science [43–45]. Together, these works position EA as essential for the coordinated and strategic integration of XAI across enterprise environments [35].

Table 2.5: Key Findings in Literature XAI

Aspect	Evaluation Metrics	Key Findings
Transparency in XAI-powered Educational Assistants [3, 10, 12]	System usability, model trustworthiness, cognitive load, interpretability ratings	RAG-based XAI chatbots improve educational engagement but require structured explanations to build trust and usability. Clearer explanations reduce cognitive load and enhance acceptance among educators.
Impact of Explainability on Student Engagement [6, 7, 27]	Student self-efficacy, problem-solving accuracy, engagement metrics	XAI chatbots tailored for interactive learning significantly improve student engagement and self-efficacy, particularly in problem-solving tasks.
Challenges in Post-hoc Explainability Methods [13, 15]	Fidelity of explanations, model accuracy before/after explanations, alignment with ground truth	SHAP and LIME often fail to produce consistent and interpretable explanations in XAI-driven educational settings. Counterfactual methods (e.g., DiCE, CEM) show higher interpretability but introduce computational overhead.

Continued on next page

Continued from previous page

Aspect	Evaluation Metrics	Key Findings
Ethical and Governance Considerations in XAI-powered Education [4, 5, 23]	Compliance with educational policies, ethical fairness, data privacy protection	XAI transparency is critical for mitigating bias and ensuring responsible XAI adoption. Standardized governance frameworks are needed to balance explainability and compliance, especially concerning GDPR and institutional regulations.
Trust and Adoption of XAI in Teaching and Administrative Roles [12, 14, 22]	Trust Scale, TAM (Technology Acceptance Model), Likert-scale user perception	Educators demonstrate higher trust in XAI systems when explanations are transparent and tailored to their expertise levels. Generic explanations lead to skepticism, whereas personalized explanations improve acceptance.
Integration of Explainable RAG Models in Learning Environments [9, 10]	Accuracy of retrieved knowledge, user understanding of XAI outputs, interactive explainability tools	RAG improves response accuracy but requires additional explanation layers to clarify knowledge provenance. Tools like RAGE improve user understanding of XAI-generated content.
Human-XAI Collaboration in Decision-Support Systems [6, 17, 28]	User trust, decision accuracy, self-explanation strategies	When instructors co-design XAI models, trust and interpretability increase significantly. Human-XAI collaboration enhances explainability without reducing system efficiency.
Algorithmic Literacy and Explainability in XAI Education [2, 21, 29]	Pre/post assessment scores, algorithmic literacy, XAI comprehension tests	Interactive XAI tools improve students' ability to interpret XAI-driven decisions, increasing overall algorithmic literacy and fostering critical thinking about XAI-generated outputs.
Scalability of XAI in Large Educational Institutions [4, 23, 26]	Model performance under different loads, compliance monitoring, XAI governance structures	Scaling XAI requires balancing model transparency with computational efficiency. Decentralized governance frameworks can help standardize explainability approaches across institutions.
Bias Mitigation and Fairness in XAI-powered Educational Systems [8, 11, 20]	Fairness assessment, algorithmic bias detection, model robustness testing	Explainability alone does not eliminate bias; explicit fairness constraints are needed within XAI-powered education tools to ensure equitable student outcomes.
Effectiveness of Self-Explanation Mechanisms for Students [16, 17]	Student engagement, accuracy in knowledge retention, comprehension tests	XAI systems incorporating self-explanation mechanisms help students understand complex decision processes, improving knowledge retention.
Adaptive Explainability for Diverse Stakeholder Groups [12, 14, 22]	Satisfaction surveys, perceived usefulness, user experience ratings	XAI-powered systems should adapt explanations dynamically based on user expertise (e.g., students, teachers, IT managers) to maximize interpretability and usability.
Trade-offs Between Transparency and Model Efficiency [13, 27, 31]	Latency, model response time, computational cost	More transparent XAI models introduce higher computational overhead. Edge computing and federated learning can mitigate performance trade-offs while maintaining explainability.

Table 2.6: Key Findings in Literature EA

Aspect	Evaluation Metrics	Key Findings
XAI Impact on EA [34]	Architectural changes, impact on business processes, technology layers	XAI adoption significantly alters various EA layers, affecting business processes, application architecture, and technology infrastructure.
Sustainability in EA for XAI Integration [36]	Policy alignment, sustainability indicators, conceptual framework evaluation	EA should align with sustainability goals and ethical XAI frameworks to ensure responsible long-term XAI adoption. Frameworks with design principles promote long-term integration.
Ethical XAI Deployment and Governance [38]	Governance models, XAI transparency, ethical XAI practices	EA structures enable transparency in XAI systems, supporting explainability and model governance, essential for ethical XAI deployment.
Digital Transformation and EA Modeling [33]	Meta-modeling, organizational impact, digital technologies	EA modeling is crucial in supporting digital transformation, helping organizations manage the impact of emerging technologies like XAI on business processes.
XAI in Smart Manufacturing [39]	XAI adoption metrics, manufacturing industry readiness, EA frameworks	EA frameworks play a key role in supporting XAI adoption in smart manufacturing, helping to integrate XAI with business processes and addressing industry-specific complexities.
Digital System Evolution and EA [40]	Technological platforms, XAI integration, digital ecosystem analysis	EA must evolve to incorporate XAI, IoT, and cloud computing, facilitating intelligent digital systems. Future EA frameworks must embrace digitalization and XAI advancements.

Continued on next page

Continued from previous page

Aspect	Evaluation Metrics	Key Findings
EA for XAI-Powered Transformation [35]	XAI-driven processes, operational efficiency, organizational change	EA facilitates XAI integration, enhancing operational efficiency and fostering innovation in XAI-powered business transformation.
TOGAF for XAI Integration [37]	XAI adoption in business, TOGAF methodology, organizational alignment	Combining TOGAF's ADM methodology with Business Architecture Body of Knowledge (BIZBOK) helps structure XAI adoption across business capabilities and value streams. Case studies demonstrate improved XAI integration.
Generative XAI Integration in EA [41]	Generative XAI, XAI adoption strategies, responsible development	Successful adoption of Generative XAI requires addressing critical factors like responsible XAI development, governance, and collaboration, aligning EA frameworks with GenAI needs.
Enterprise Analytics Strategy [42]	Data-driven analytics, cloud platforms, XAI solutions	A strategic blueprint combining EA and analytics fosters growth by optimizing data management, XAI solutions, and business strategy. Case studies show success in large enterprises.
XAI Solution Development in EA [43]	Requirements engineering, XAI context analysis, organizational readiness	EA models support XAI requirements engineering by assessing organizational readiness and providing context analysis for different XAI application types.
XAI System Integration in EA [44]	Data science integration, collaboration in development cycles, XAI system efficiency	XAI system integration within EA benefits from best practices in data science and DevOps, facilitating efficient governance and collaboration across development cycles.
XAI Service System Development with EA [45]	XAI service systems, EA modeling techniques, system scalability	EA modeling helps design scalable XAI service systems that integrate with organizational frameworks, ensuring alignment with business processes.

2.3.1. SETTINGS ANALYSIS

A settings analysis is conducted to identify the specific contexts and environments where XAI integration can most effectively support educational goals. The literature highlights a broad range of settings where XAI is integrated into educational systems. Higher education and K-12 [5] environments are the most prominent, with XAI-powered tools assisting educators and students. Hennekeuser and Kasneci [10, 27] examine the role of LLMs in providing instructional support, automated grading, and real-time feedback. These studies present how XAI-driven solutions can augment traditional teaching methods by offering personalized feedback and real-time assistance to enhances learning efficiency [14]. The strategic deployment of such tools benefits from EA frameworks that guide how XAI systems are integrated across pedagogical, application, and infrastructure layers, ensuring alignment with institutional goals [38, 40]. Furthermore, integrating XAI frameworks within educational environments can significantly improve decision-making and enhance pedagogical goals, especially when aligned with EA strategies designed to optimize XAI-driven educational tools through structured models that incorporate explainability into enterprise-wide systems [35, 37].

Additionally, the ability to explain XAI-driven predictions in MOOCs is particularly crucial as these platforms serve diverse learners with varying levels of digital literacy [13]. XAI explanations also influence teachers' trust and technology adoption in STEM classrooms, demonstrating that transparent XAI models improve instructional decision-making and classroom interactions [12]. In such context, EA can offer an architectural blueprint for embedding explain-

[5] K–12 environments refer to formal educational settings from kindergarten through 12th grade, often explored in XAI and XAI integration studies for their unique instructional and developmental contexts [5, 27].

ability mechanisms within learning platforms, ensuring both scalability and interoperability with existing educational technologies [41, 43]. The successful adoption of XAI technologies in educational platforms can benefit greatly from EA frameworks that promote scalability and the integration of emerging XAI-driven tools in business processes [43] while addressing requirements for compliance, data governance, and system adaptability [33, 34].

Further emphasis is placed on XAI and its alignment with IT governance frameworks. Significant importance is placed on ensuring XAI-driven decision-making remains transparent and accountable [23]. Compliance with educational policies and institutional regulations is a recurring challenge in large-scale XAI deployments, highlighting the importance of explainability in mitigating bias and ensuring fair adoption [5, 29]. EA frameworks play a pivotal role in aligning XAI governance with educational policy, providing a transparent structure for decision-making that is ethical and sustainable by incorporating policy-aligned modeling, stakeholder collaboration, and lifecycle-based oversight for responsible XAI integration [36, 41].

2.3.2. TECHNIQUES ANALYSIS

An analysis of XAI techniques is essential to understand how different methods enhance model transparency, interpretability, and applicability in educational contexts. The techniques employed in the reviewed literature shopwcase diverse range of XAI methodologies. Methods such as SHAP, LIME, and PRISMA are used in XAI-driven educational applications [8, 15, 21, 23]. These techniques provide insights into XAI model decisions by identifying feature importance, making them essential tools for ensuring transparency in complex predictive models. Furthermore, EA frameworks can support the integration of such XAI techniques into organizational strategies, ensuring both technical and operational alignment [34, 40].

Another critical approach is RAG, which enhances XAI chatbots by integrating external knowledge sources to improve response accuracy [9, 10, 31]. RAG-powered assistants can enhance lecturer support while maintaining transparency [10]. Additionaly, EA integration with these models, help ensure consistency and scalability, as they access real-time, contextually relevant information for learners and instructors alike [37],

Furthermore, RAG allows XAI models to dynamically access up-to-date information, making them more adaptable and informative than traditional static models [9]. EA models designed for educational institutions can incorporate dynamic learning environments powered by such RAG-driven tools, further enhancing their utility in both classroom and online learning settings. [43].

Neural-symbolic hybrid models also emerge as a promising direction, combining deep learning techniques with symbolic reasoning to enhance interpretability [13, 15]. From an EA perspective, these hybrid models are a perfect fit for systems that require both the flexibility of XAI and the structured decision-making process essential for educational institutions [39].

Additionally, interactive human-XAI collaboration models enable real-time adjustments to XAI-

generated predictions [6], improving explainability and user trust. By involving human instructors in XAI model refinement, these approaches ensure that XAI explanations remain contextually relevant and aligned with pedagogical goals [11, 28]. Such integration ensures that the XAI systems remain flexible within the dynamic environment of modern educational systems [45].

2.3.3. EVALUATION METHODS ANALYSIS

Evaluating explainable AI (XAI) methods is essential to ensure their transparency, effectiveness, and alignment with user and organizational needs. Evaluation methods in XAI research employ a diverse range of metrics to assess XAI model transparency and effectiveness [27, 29]. For instance, many studies use quantitative performance metrics, such as accuracy before and after explainability integration [15].

User trust and acceptance evaluations are also widely employed [12, 22]. Technology Acceptance Model (TAM) and trust scales can be used to measure stakeholder confidence in XAI-powered classroom tools [12]. The literature suggests that clarity of XAI explanations significantly influences educators' willingness to adopt XAI-driven analytics platforms. Moreover, different explainability techniques can yield varying levels of user trust, reinforcing the need for tailored approaches depending on the audience [13].

In parallel, studies within EA domain emphasize the importance of structured evaluation techniques to assess the alignment between XAI explainability and organizational frameworks [36]. EA-driven approaches help contextualize explainability methods within business processes, ensuring that transparency metrics are not only technically sound but also strategically relevant [38, 40, 43]. Theoretical evaluations in EA studies often rely on literature synthesis, conceptual frameworks, or qualitative justification to assess the impact of XAI integration on architectural layers and governance outcomes [34, 36].These evaluations often incorporate EA models to determine how XAI components interact with enterprise layers, such as application and business architecture, thereby supporting more holistic assessments that consider both technical feasibility and strategic coherence [35, 40].

Additionally, engagement and learning outcome assessments are also used in interactive XAI environments [7]. Qualitative evaluations, including expert reviews and structured interviews, are also used to assess XAI compliance with governance frameworks [4, 17]. This aligns with EA methods that include case study-driven evaluations and scenario testing to validate how XAI systems contribute to organizational change and decision support [33, 39]. These studies highlight that beyond technical accuracy, user perception of XAI transparency plays a significant role in determining adoption rates in educational settings. Borrowing from EA's evaluation logic XAI assessments can better align with enterprise-level goals and ensure systemic readiness [37, 43].

2.4. DISCUSSION

The following section synthesizes key findings from the literature, examining the integration of XAI within educational systems through architectural, technical, and governance lenses. Integrating XAI into educational environments requires modular architectures that balance transparency and efficiency. RAG frameworks enhance XAI reliability while preserving interpretability [10]. Additionally, human-in-the-loop mechanisms ensure XAI models remain adaptable and context-aware in educational settings [6]. By involving educators in model updates and refinements, these approaches allow for continuous improvements in chatbot accuracy and explainability.

Building on the theme of trust and usability, we next turn our attention to how explainability affects stakeholders. Transparency and interpretability are central to building trust in XAI-powered education [12]. Tailored explanation strategies, such as adaptive self-explanation techniques [17], bridge the knowledge gap between XAI models and diverse stakeholders. Visual represetation tools, like RAGE [9], further improve interpretability for users with varying levels of XAI expertise by offering graphical breakdowns of model reasoning.

Institutions, however, must prioritize compliance. Ensuring compliance with educational governance frameworks remains a challenge. Research underscores the importance of fairness, accountability, and transparency in XAI adoption [4]. However,caution is advised when considering trade-offs between model performance and interpretability [13]. The scalability of XAI techniques also remains a pressing concern, particularly as XAI-powered solutions expand to accommodate larger user bases.

Scaling XAI across institutional IT systems presents additional risks, particularly in maintaining data privacy and standardizing transparency practices [4]. Addressing these challenges requires developing flexible XAI governance models that align with evolving compliance requirements [23]. By implementing explainability standards that integrate with regulatory frameworks, educational institutions can foster ethical XAI adoption while minimizing operational risks .

Lastly, we examine the impact of XAI at a structural level, focusing on EA within educational institutions. Rittelmeyer highlighted that XAI applications can lead to significant changes in a company's EA, including data formats, application landscapes, and business processes [34]. Understanding these impacts can help educational institutions anticipate and manage changes effectively when integrating XAI solutions.

The research by Rittlemeyer [34] suggests that EA models can be instrumental in evaluating organizational preparedness for XAI adoption by analyzing existing structures and identifying necessary adjustments. Furthermore, EA patterns play a critical role in enabling XAI system integration in a sustainable and adaptable way [36, 38]. These patterns can be mapped to support explainability through clear modularization of logic, traceable data flows, and separation of concerns—key elements for building transparent and auditable XAI systems in educational contexts.

2.5. MANAGERIAL IMPLICATIONS

2.5.1. EFFECTIVE INTEGRATION OF TOOLS IN EDUCATION

To effectively integrate LLM chatbots within an educational environment, focus must be put on structured implementation, collaboration, and continuous adaptation, see the outlined framework for implementation in A.6:

The initial step involves aligning XAI-powered educational assistants align with institution's technological objectives. This involves considering the current IT infrastructure, pedagogical approaches, and the specific needs of faculty and students. [45, 46]. Leveraging EA frameworks can support this alignment by offering guidance on strategic technological fit [47].

Building trust is crucial. Transparency is crucial to gaining the trust of both educational staff, students,and faculty. Universities should prioritize the adoption of transparent XAI models [48] and ensure that explanations are comprehensible to users with varying levels of knowledge [49]. Institutions can improve understanding through structured communication strategies and stakeholder education around how XAI systems work and how outputs should be interpreted [49].

Institutions must actively preserve both performance and explainability as they scale systems across departments or campuses. This involves selecting EA approaches that support flexibility and growth [44, 47], ensuring that the XAI system can evolve in response to changing educational needs.

TOGAF offers a comprehensive methodology with the Architecture Development Method (ADM), which can be adapted to educational environments[46], ensuring flexibility and scalability in implementation. Similarly, the Zachman Framework (ZF), while more abstract, emphasizes the need for clear perspectives from different stakeholders. [50, 51]. See Table 2.7 for a comparison of the most popular EA frameworks and their strengths and weaknesses.

Table 2.7: Comparison of Enterprise Architecture Frameworks: Strengths, Weaknesses, and Evaluation Metrics

Framework	Strengths	Weaknesses	Evaluation Metrics
TOGAF [52]	Comprehensive methodology with the ADM; widely adopted with extensive support; flexible and adaptable to various organizational needs.	Can be complex and resource-intensive; may be too theoretical without practical tailoring; requires significant training and expertise.	Adaptability, scalability, support for modern technologies, cost-effectiveness, compliance, traceability, risk management.
Zachman Framework [53]	Provides a structured taxonomy for organizing architectural artifacts; emphasizes different stakeholder perspectives; strong theoretical foundation.	Lacks a defined process for implementation; can be abstract and challenging to apply practically; limited guidance on execution.	Adaptability, scalability, ease of implementation, support for modern technologies.
FEAF [54]	Standardized approach promoting consistency and interoperability; aligns IT investments with business goals; emphasizes performance measurement.	Primarily tailored for U.S. federal agencies; may lack flexibility for diverse organizational contexts; implementation can be resource-intensive.	Governance guidance, alignment with federal regulations, performance measurement, inter-agency collaboration.

Framework	Strengths	Weaknesses	Evaluation Metrics
Gartner Framework[55]	Business-outcome-driven approach; focuses on aligning IT with business strategy; emphasizes understanding and enabling business outcomes.	May not provide comprehensive architectural guidance; could require integration with other frameworks for full implementation.	Business focus, governance guidance, partitioning guidance, prescriptive catalog.
DoDAF [56]	Structured approach with standardized views and models; emphasizes data-centric architecture; suitable for complex, large-scale systems.	Primarily designed for defense and government contexts; may be too rigid for commercial applications; steep learning curve.	Interoperability, standardization, data-centric modeling, compliance with defense standards.
Integrated Architecture Framework [57]	Covers business, information, systems, and technology domains; integrates governance, security, and sustainability; adaptable to various organizational contexts.	Less widely adopted compared to other frameworks; may require customization for specific needs; limited public documentation.	Alignment with business goals, comprehensive domain coverage, adaptability, integration of governance and security.
Dragon1 [58]	Emphasizes visual architecture products for strategic decision support; promotes architecture as a total concept design discipline; supports stakeholder engagement through visualizations.	Relatively new and less established; may require integration with other frameworks; limited adoption and community support.	Visual modeling, stakeholder engagement, strategic alignment, support for enterprise transformations.

As outlined in Table 2.7, TOGAF is widely adopted and flexible but can be resource-intensive. ZF is valuable for organizing architectural artifacts but lacks practical implementation guidance [51, 59]. FEAF is standardized and promotes interoperability, though it is more suited for larger organizations. focuses on aligning IT with business outcomes but may require integration with other frameworks for full implementation. Other frameworks, like DoDAF and IAF, offer specialized advantages but may not be as adaptable for educational institutions. Dragon1 supports strategic decision-making through visualizations but faces adoption challenges [59, 60].

Martel et al. [44] focused on best practices for integrating AI into enterprise software architecture, providing case studies and theoretical analysis that demonstrate the practical applications of EA frameworks in AI deployment. Similarly, Takeuchi and Yamamoto [45] proposed an EA-based modeling approach specifically for AI service systems, offering conceptual modeling tools that can align AI services with organizational processes [45]. Similar studies, summarized in Table A3, provide insights into comparing EA frameworks employed to enterprise contexts in terms of purpose and technique. [50, 59, 60]

Integration of XAI into educational environments must adhere to institutional governance frameworks, including data privacy regulations (i.e. GDPR), and ethical standards. XAI systems ought be designed with these requirements in mind, to maintain transparency and accountability [48, 49].

2.5.2. REGULATORY AND ETHICAL USAGE OF TOOLS IN EDUCATION

The adoption of XAI in education brings significant regulatory and ethical challenges. Compliance with data privacy laws such as GDPR [6] and EU XAI [7] act to ensure student data is securely collected, stored, and processed. As XAI evolves, regulatory frameworks must adapt to address challenges like algorithmic decision-making and automated data processing, safeguarding sensitive information and preventing unauthorized access.

Ethically, the system must be fair, and transparent, as described by the EU commission's ethics guidelines [8]. XAI models trained on biased data can perpetuate inequalities, making regular audits necessary to ensure fairness. Transparency is also crucial—students, educators, and administrators should understand how XAI-driven decisions, and support in personalized learning, are made.

XAI should enhance, not replace, human decision-making in education. While automation can improve learning experiences, it must not undermine critical thinking or student autonomy. Ethical XAI use requires a balance between automation and human oversight [6, 28]. Additionally, institutions must ensure alignment with ethical standards and regulatory requirements.

2.6. CHAPTER SUMMARY

The literature review emphasizes that transparency and interpretability are crucial for fostering trust among diverse educational stakeholders, who benefit from tailored explanation methods. Adaptive explainability techniques, such as self-explanations and interactive visualizations, show promise for enhancing user engagement, though their large-scale application remains underexplored. Aligning XAI with IT governance and privacy regulations like GDPR is essential but challenging, calling for standardized frameworks. Despite growing theoretical insights, empirical validation of explainability's impact on learning outcomes is limited. Additionally, balancing transparency with computational efficiency and scalability poses significant challenges, with emerging solutions like edge computing and federated learning requiring further research. The literature also warns of algorithmic bias risks, especially in models pretrained on biased data.

[6] GDPR as in the European Union (EU) law that governs how personal data is collected, processed, and stored, aiming to protect individuals' privacy

[7] XAI act as in the proposed by EU regulation that aims to ensure safe and trustworthy use of artificial intelligence in the EU by setting rules based on the level of risk posed by different XAI systems.

[8] Ethics guidelines for trustworthy XAI

3

METHODOLOGY

The following chapter embarks on a comprehensive analysis of the methodologies employed in the current study. Methodology serves as the guiding framework that shapes the entire research process, ensuring clarity, and effectiveness in achieving the study's objectives. The following chapter is divided into four distinct sections: the DSR, the CRISP-ML, the MODES-EA , and the Analytical Methods sections. By dividing the methodology chapter into four separate subsections, the study aims to provide a complete overview of the whole research framework.

3.1. DESIGN SCIENCE RESEARCH

DSR is a methodology focused on the iterative creation and evaluation of artifacts [61]. Following the DSR structure, we obtain a six-staged cycle: problem identification, objective definition, design and development, demonstration, evaluation, and communication (see Figure B.1 and Table 3.1). Each stage builds on the previous, allowing iterative refinement and validation.

Table 3.1: DSR Cycle for XAI Integration in EA

DSR Stage	Description	Purpose	Research Lens	Outputs / Measures
Identify Problem and Motivate	Identify challenges related to XAI integration within EA frameworks.	Understand the key issues and justify the need for XAI within the context of EA.	Problem Definition	Problem statements and motivations for addressing XAI challenges in EA frameworks.
Define Objective of a Solution	Set clear objectives for integrating XAI into EA setups based on the identified challenges.	Establish the goals for what the solution should achieve (e.g., transparency, explainability).	Solution Objectives	Clear goals and metrics for evaluating XAI integration success.
Design and Development	Develop the solution architecture, including choosing XAI techniques, frameworks, and tools to meet objectives.	Create the technical solution, considering the constraints and needs of EA frameworks.	Design and Development Theory	Prototypes or working solutions for integrating XAI into EA setups, including technical specifications.
Demonstration	Demonstrate the developed solution through trials or use cases.	Validate the solution's effectiveness and performance in real-world scenarios.	Experimental Research	Results from solution trials, case studies, or pilot tests showing effectiveness and potential limitations.

DSR Stage	Description	Purpose	Research Lens	Outputs / Measures
Evaluation	Assess the solution's performance based on defined objectives and real-world feedback.	Evaluate the solution's alignment with objectives, including trust, transparency, and compliance.	Iterative Evaluation	Evaluation reports on solution performance, user feedback, and improvements based on real-world testing.
Communication	Communicate the findings, results, and recommendations from the project to stakeholders.	Share results, lessons learned, and best practices with the wider community.	Knowledge Transfer	Publications, reports, and presentations summarizing the solution, results, and recommendations for XAI integration.

DSR provides a clear methodology to follow while designing artefacts (see Chapters 4 and 5), simulating, and comparing EA models and their impact on XAI. This process ensures that solutions evolve based on empirical evidence rather than assumptions, enhancing their relevance and effectiveness.

DSR ensures alignment between technical design and organizational goals. Said alignment ensures that the developed solutions not only meet technical specifications but also address stakeholder needs and regulatory requirements. Such approach led to the development of two artifacts (Chapters 4 and 5: 1) an educational chatbot integrating RAG and prompt engineering withinLearning Management System (LMS)s, (see Chapter 4); 2) Flowchitect, a tool for modeling, simulating and evaluating ArchiMate models (see Chapter 5).

3.2. CROSS-INDUSTRY STANDARD PROCESS FOR MACHINE LEARNING WITH QUALITY ASSURANCE

The study adopts CRISP-ML as its second methodological framework. CRISP-ML provides a structured, approach for developing machine learning systems (see Figure B.2). It guides the investigation of RAG effects on explainability and factual alignment in LLM-based educational chatbots, tailoring each phase to technical and interpretability goals (see Table 3.2).

By applying CRISP-ML, the study ensures systematic experimentation, reproducibility, and consistent evaluation. Table 3.2 details how each phase supports the research objectives.

Table 3.2: CRISP-ML process stages for the XAI in educational LLM chatbots experiment

Phase	Component	Objective	Details/Actions	Outputs / Measures
Business & Data Understanding	Research Goal	Investigate the effect of RAG on explainability and source alignment in educational chatbots	Compare standard LLM and LLM+RAG for clarity, transparency, and factual consistency with provided course data	Defined evaluation focus: explainability, traceability, and alignment
	Success Criteria	Establish clear indicators of improved explainability	Focus on traceability to source material, semantic accuracy, and user trust	Mix of qualitative and quantitative XAI metrics
	Data Sources	Identify and prepare course materials and user prompts	Use structured educational content and create representative questions	Cleaned corpus and standardized prompts
Data Preparation	Corpus Indexing	Prepare content for RAG pipeline	Clean and structure content for semantic retrieval	Indexed corpus for relevant access
	Input Engineering	Ensure input consistency	Format queries with aligned user intent	Controlled inputs for comparison
Modeling	LLM Baseline	Test general-purpose LLM without retrieval	Use API with zero/few-shot prompting	Raw responses stored
	LLM + RAG	Enhance response with retrieved data	Embed and retrieve course segments	RAG-enhanced outputs

Phase	Component	Objective	Details/Actions	Outputs / Measures
Evaluation	XAI Metrics	Measure explainability, traceability, clarity	Use human/rubric scoring: source highlighting, coherence, confidence	Traceability scores, subjective ratings
	Accuracy Metrics	Assess factual consistency	Focus on semantic fidelity	Alignment scores with ground truth
Deployment	Optional Interface	Prototype for interaction/testing	Simple UI for feedback and comparison	Usability feedback (optional)
Monitoring	Iteration Planning	Enable iterative improvement	Track explanation failures and refine logic	Improved iterations

CRISP-ML is well-suited iterative experimentation with RAG configurations, supporting systematic measurement of transparency, traceability, and factual alignment in LLM outputs. This aligns with DSR goals of artifact construction and evaluation.

3.3. SIMULATION OF ENTERPRISE ARCHITECTURE MODELS (MODES-EA)

The MODES-EA is introduced in this thesis as a novel, structured methodology specifically designed to formalize and guide the development of Discrete-Event Simulation (DES) studies (see Figure 3.1). As the first articulation of this framework, MODES-EA provides conceptual clarity, and procedural consistency in how DES simulations are designed, executed, and analyzed. It is based on, and extends, the Layered MDE framework originally presented by Wilsdorf et al. [1] (Figure B.3), see Table B.1 for more detailed overview).

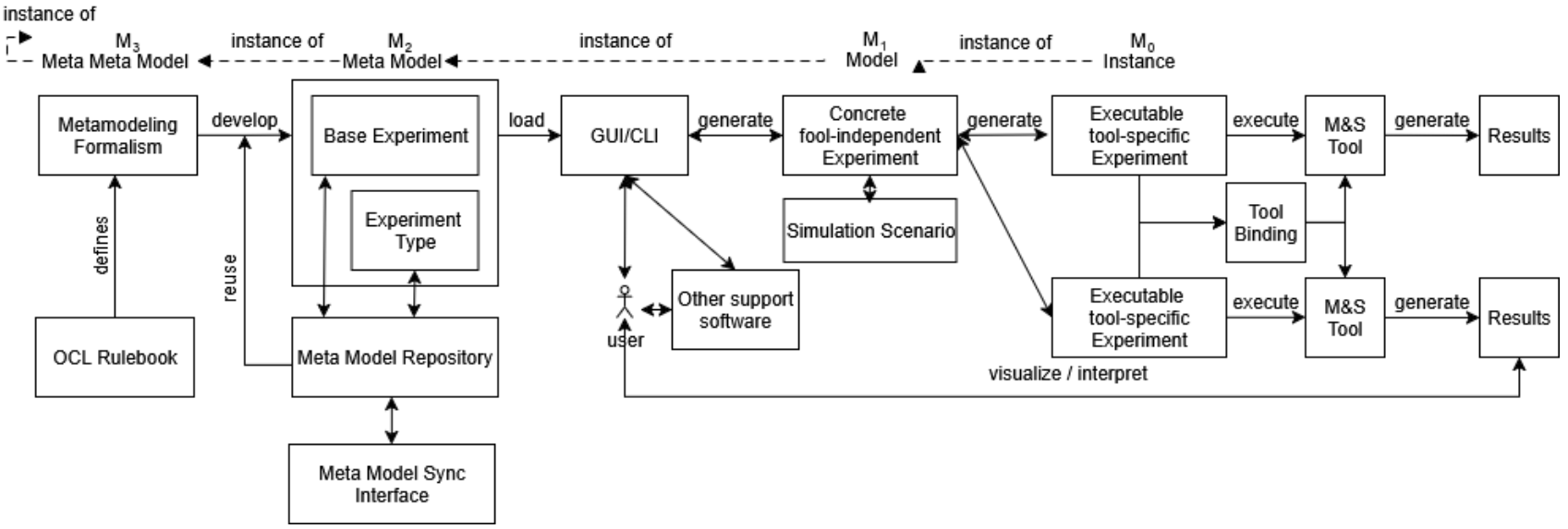


Figure 3.1: MODES-EA Simulation Methodology

Based on this information, we propose an improvement to the framework to more effectively address the specific requirements of DES in the context of this study. Similar to its predecessor, MODES-EA retains the four modeling layers but enhances them to address the complexity of EA-specific simulation:

At the M3 level (see Table B.2), the meta-meta model is expanded with constructs that explicitly support multi-layered EA frameworks. New meta-meta classes capture hierarchical relationships across business, application, and technology domains, and formal constraints are introduced to govern valid cross-layer dependencies .

The M2 level (see Table B.3) is refined to incorporate domain-specific modeling elements. It introduces constructs such as architectural components, applications, data stores, infrastructure nodes, and detailed process flows that account for control and data dependencies. Roles are modeled with attributes that affect task prioritization and resource availability. Additionally, elements like queues and event triggers allow simulation of complex component interactions within EA environments.

At the M1 level (Table B.4, the abstract meta-model is instantiated to represent concrete EA scenarios. These models define workflows with branching decision logic, simulate application servers with variable processing capabilities, and use empirical parameters such as transaction volume and user load to configure scenarios. This enables the simulation to reveal performance bottlenecks, evaluate system throughput, and assess the operational impact of architectural changes).

Finally, the M0 level (see Table B.5) focuses on generating the executable artifacts that run on simulation platforms. This includes the transformation of M1 models into simulation code, along with standardized interfaces to enable compatibility with multiple simulation engines. Protocols for capturing metadata—such as runtime logs, parameter configurations, and performance metrics—are also defined. These improvements facilitate automated scenario execution, support rigorous experiment management, and ensure full traceability between models and simulation outcomes.

Furthermore, the model is updated by establishing modeling practices [62–64], structured experiment design [65, 66], and validation methodologies [67–69], while introducing a dedicated process flow that supports modular abstraction, tool-independent implementation, and reproducibility.

Structured modeling practices promote the explicit separation between logical, conceptual, and implementation layers, as emphasized in DES modeling principles [62]. These layers enable selective experimentation, such as altering customer queuing behavior in the logical layer while maintaining system deployment constraints in the implementation layer [64]. Tool-independent implementation is supported through abstract model design and standardized model exchange formats, ensuring models remain decoupled from simulation software environments [63].

Reproducibility is achieved by adhering to experimental design protocols, such as random

number stream control, replications, and output analysis procedures that ensure statistically consistent results across runs [65, 66].

In MODES-EA traceability is operationalized through pattern-based traceability modeling, where recurring architectural fragments are documented and verified against process definitions [70]. These patterns serve as reusable traceability templates, streamlining both the model construction and validation process [71].

As a methodology coined and developed in this thesis, MODES-EA serves as both a theoretical contribution and a practical foundation for conducting well-structured DES simulation experiments for EA models. To ensure credibility, the methodology aligns with best-practice validation techniques such as face validation, historical data comparison, and sensitivity analysis [68]. Additionally, statistical validation techniques are applied to assess output fidelity, see Tables B.6 & B.7.

3.4. ANALYTICAL METHODS

3.4.1. EVALUATING RAG WITH THE RAGAS FRAMEWORK

One of the core components of the analytical strategy adopted in this study, the RAGAS framework [72], provides a structured and scalable approach to evaluating the performance of RAG systems. Positioned within the broader CRISP-ML and DSR frameworks, RAGAS allows for score-based evaluation, directly tied to explainability, source traceability, and factual alignment [8, 11, 27]. Its modular structure and emphasis on semantic and contextual metrics directly support the study's aim to compare the outputs of baseline LLMs against those enhanced with RAG capabilities in educational chatbot scenarios [10].

RAGAS addresses the inability to disentangle and individually assess the retrieval and generation components of a RAG pipeline [19, 21]. To resolve this, it independently assesses the quality of retrieved context and the groundedness of the generated response. The framework itself is built around a visual scoring system that divides the evaluation into two main sections: Generation and Retrieval, see Figure 3.2 for detailed overview.

Generation

- **Faithfulness:** Measures how factually accurate the generated answer is, ensuring that the response remains true to known facts and does not introduce misleading or incorrect information.
- **Answer Relevancy:** Quantifies how well the AI's response addresses the original query, ensuring that it is both complete and contextually relevant.

Retrieval

- **Context Precision:** Assesses the signal-to-noise ratio of the retrieved context, ensuring that the AI retrieves useful and relevant information with minimal irrelevant data.
- **Context Recall:** Measures the AI's ability to retrieve all the relevant information needed to fully answer the question, ensuring comprehensive coverage of the topic.[6, 13]

Figure 3.2: RAGAS Framework Evaluation Criteria

This structure ensures independent evaluation of retrieval and generation components, with clear focus on the relevance of retrieved context and factual consistency [5, 12, 15, 20]. The RAGAS score thus provides a comprehensive measure of system performance, emphasizing both data retrieval quality and generation accuracy [21, 29].

Central to the framework is its use of model-based semantic embeddings and natural language inference to assess answer quality, enabling evaluation without the need for manually annotated references, which is proven to be a critical advantage for domains where multiple correct answers exist [3, 4, 14].

Additionally, RAGAS facilitates diagnostic analysis by identifying discrepancies between retrieval and generation metrics, allowing precise localization of performance bottlenecks within the pipeline [25, 31]. Important to note is While highly effective in technical and semantic evaluation, RAGAS does not address user-centric factors such as usability, satisfaction, or cognitive load.

3.4.2. EVALUATING RAG WITH THE RAGE FRAMEWORK

To complement automated metrics and provide deeper insight into trustworthiness and groundedness in RAG systems, we incorporate the RAGE framework by Rorseth et al. [9], which enables human-centered, provenance-based explanations via counterfactual context pruning. While RAGAS automates faithfulness and relevance scoring, RAGE enhances trust in model outputs by identifying the minimal retrieved evidence necessary for a given response, thereby improving transparency and answer traceability in LLMs [2, 21, 30].

We structure our evaluation around three dimensions supported by the attribution mechanisms of the RAGE framework (see below). This decomposition draws from established principles in explainable and trustworthy AI [11, 20], and while not formally defined by RAGE, aligns well with its multi-level attribution strategy.

Input Faithfulness (IF) assesses whether the retrieved documents are relevant and necessary to answer the query, promoting semantic alignment and reducing spurious context [6, 26]. This is especially crucial in domains such as education, where off-topic or misleading information undermines user trust. Counterfactual analysis validates the necessity of retrieved content [20].

Output Faithfulness (OF) evaluates whether the generated response remains grounded in the retrieved inputs, ensuring that the model does not hallucinate or misrepresent facts [5, 10]. This is typically assessed via entailment models, fact-checkers, or sensitivity tests [22, 29].

Source Attribution (SA) measures how accurately and clearly the model references its sources, a critical factor in user confidence and cognitive ease [4, 21]. Attribution quality can be evaluated through manual annotation or automated citation relevance checks [14].

Unlike fully automated pipelines, the RAGE framework prioritizes interpretability and user-facing trust, making it well-suited for rigorous, human-in-the-loop evaluation scenarios [8, 25]. It also supports proxy metrics: semantic alignment and necessity tests for IF, contradiction de-

tection for OF, and rubric-based citation scoring for SA [19, 23]. RAGE complements CRISP-ML by enriching evaluation with trust-centered diagnostics, and aligns with DSR through its focus on verifiable, user-oriented system behavior [5, 16]. Its multi-level attribution capabilities—document, sentence, and span—support iterative model refinement aligned with trustworthy AI principles [7, 29].

3.4.3. EVALUATION OF EA MODELS

Our evaluation follows the methodology by Sandkuhl and Rittelmeyer [34, 43], which defines key steps to assess EA models for AI solution development (Table B.8). We adapt this by focusing on classifying AI application types .

Different AI applications have distinct requirements for data, computation, and organization [36, 40]. Evaluating EA components involves analyzing their ability to support scalability and flexibility needed for AI [35, 43, 44].

Next, we align EA capabilities with AI requirements through requirements engineering, ensuring integration into business processes [38, 41]. The final step assesses organizational readiness in terms of culture, technology, and structure to enable AI adoption [39, 47, 49].

COVERAGE METRICS

Layer Coverage Ratio (LCR) quantifies how completely an EA model captures essential business and IT elements [47]:

$$LCR = \frac{\text{Number of modeled elements}}{\text{Total possible elements}}$$

Traceability Ration (TR) measures the extent of traceability between elements across architectural layers, supporting consistency and impact analysis [47]:

$$TR = \frac{\text{Number of traceable links}}{\text{Total possible links}}$$

ALIGNMENT METRICS

Business–IT alignment gauges how well IT supports business goals [73, 74]. The Business-IT Alignment Index (BITAI) index provides a normalized score for this alignment:

$$BITAI = \frac{\sum_{i=1}^{n} w_i \cdot s_i}{\sum_{i=1}^{n} w_i}$$

where s_i are alignment factor scores and w_i their weights, typically derived from frameworks such as Luftman's Strategic Alignment Maturity Model (SAMM) [73]no .

Additionally, Goal Satisfaction Index (GSC) assesses the proportion of organizational goals supported by architectural elements [47]:

$$GSC = \frac{\text{Number of goals supported}}{\text{Total goals}}$$

These metrics together provide both quantitative and qualitative measures of EA effectiveness in aligning business and IT strategies and supporting AI integration.

3.4.4. EVALUATION OF SIMULATION

Qualitative evaluation (interviews, structured argumentation, and credibility evaluation)[66, 67, 75] in EA simulations focuses on assessing conceptual fidelity, internal coherence, and the perceived credibility[1, 68, 71, 76] of the simulation model [64, 67]. This involves face validation, where domain experts review the model's structure, assumptions, and logic to ensure plausibility [65, 67–69]. Model review breaks down the simulation into components, verifying conceptual validity by checking assumptions, boundary conditions, and abstraction levels [63, 77].

Static structure analysis complements this by detecting logical errors, redundancies, and completeness of control flows. The iterative Verification, Validation, and Accreditation (VVA) lifecycle supports ongoing verification and validation throughout development, ensuring the model is both correctly implemented and fit for purpose [68, 69].

Quantitative evaluation complements qualitative approaches by mathematically verifying accuracy, predictive power, and fidelity of simulation outputs against observed data and theoretical expectations [67, 68]. This ensures the model produces reliable results suitable for decision-making.

ERROR AND ACCURACY METRICS

Key error metrics include Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE), which measure absolute and squared differences between simulated and observed values. Normalized metrics like Normalized Mean Squared Error (NMSE) provide scale-independent accuracy by relating squared errors to the variance in observed data, allowing comparisons across different models or outputs [65]. Theil's U Statistic (TheilU) offers a relative forecast accuracy measure by comparing root mean squared errors to the magnitude of observed data[69]

CONSISTENCY AND REPEATABILITY CHECKS

R^2 Coefficient of Determination (R2) measures the proportion of variance in observed data explained by the simulation, serving as a key indicator of model fit and predictive quality [65]:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

4

TRUSTWORTHY AI

This chapter presents a detailed description of the experimental procedures used to evaluate different retriever methods within the study. The evaluation focuses on three key retrieval techniques: BM25, cosine similarity, and a hybrid approach. To assess their effectiveness, the RAGAS and RAGE frameworks are employed as benchmarking tools.

4.1. DATA COLLECTION AND DESCRIPTION

This section outlines the nature and structure of the data collected for analysis, focusing on three core types of academic content: summaries, lectures, and textbooks. These data types are to be extracted from educational platforms with an emphasis on university-level theoretical materials. Given their heterogeneous formats and diverse cognitive functions, a comparative schema is introduced to characterize them along dimensions considered relevant by the study.

4.1.1. DATA PRIMITIVES

Table 4.1 provides an overview of three principal categories of educational content extracted from academic platforms: summaries, lectures, and books. The data types are assessed across multiple dimensions of interest, including structural complexity, information density, metadata richness, and annotation potential.

Table 4.1: Characterization of Extracted Educational Data Types

Data Type	File Types	Avg. Size	Structural Complexity	Information Density	Parsing Requirements	Metadata Availability	Annotation Potential
Summaries	`.pdf`, `.docx`, `.txt`	~9mb	Low	High	Low	Low	High
Lectures	`.pdf`, `.docx`, `.pptx`	~20mb	High	Medium	High	Medium	Medium
Books	`.pdf`	~90mb	High	High	Medium	High	High

Where Structural Complexity refers to the degree of internal organization and formatting embedded within a data item. This includes hierarchical nesting (e.g., sections, chapters), multimedia layering, and formal semantic divisions. Higher complexity often implies greater expressiveness but also increased difficulty in parsing and segmentation. Table C.1 below defines the operational range of this dimension.

Closely related, Information Density captures the conceptual saturation within a given unit of data—typically expressed as the volume of theoretical, factual, or analytical content per textual or temporal segment. Table C.2 provides descriptors for the qualitative levels of this attribute.

Parsing Requirements evaluate the computational effort and sophistication necessary to transform raw data into machine-readable formats. This includes tasks like layout analysis, structure recognition. Table C.3 distinguishes between data that can be trivially extracted and that which requires specialized pipelines.

Metadata Availability measures the extent to which embedded or associated metadata is present, structured, and machine-accessible. Metadata supports indexing, classification, filtering, and search, and is a strong enabler for reproducible data pipelines. Table C.4 describes levels of availability typically encountered across academic data types.

Finally, Annotation potential assesses the ease with which a data type can be manually or automatically segmented and labeled for information retrieval, or domain-specific tagging. This dimension is particularly relevant for creating gold-standard datasets and performing fine-grained analysis. Table C.5 defines annotation feasibility across qualitative levels

4.1.2. DATA RELATIONSHIP

Figure 4.1 illustrates the hierarchical structure used to organize the educational dataset. At the top level, a Course represents a distinct academic unit, typically aligned with a university syllabus.

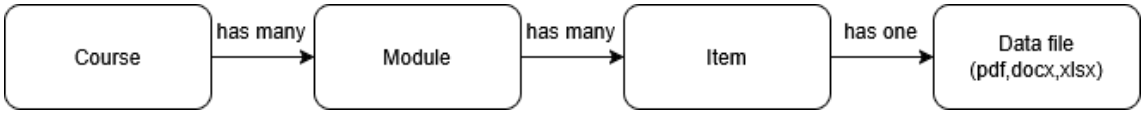


Figure 4.1: Data Description - Relations between elements

Each course is subdivided into multiple Modules, reflecting thematic or chronological subdivisions (e.g., weeks or topics). Within each module, multiple Items are defined, corresponding to concrete learning resources such as lectures, summaries, readings, or exercises. Each item is associated with exactly one Data File, representing the educational digital asset (e.g., .pdf, .docx, .pptx, .xlsx).

A Course represents the highest organizational level in the academic content hierarchy. It encapsulates a comprehensive body of knowledge aligned with specific university curricula or academic programs. Each course is designed to achieve defined learning objectives and outcomes, providing a structured pathway through theoretical and practical material. Courses are

typically identified by unique codes or titles and encompass multiple modules that segment the content into manageable thematic or temporal units, see Table C.6.

A Module functions as a subdivision within a course, organizing content around focused topics, weeks, or thematic units. Modules aggregate various instructional items, ensuring logical sequencing and pedagogical coherence (see detailed structure in Table C.7). They often correspond to natural breaks in the curriculum, such as chapters, weeks, or units, enabling both students and educators to track progress and manage workload.

An Item refers to an individual educational resource or learning artifact within a module. This can include lectures, summaries, readings, problem sets, or multimedia content. Each item serves a specific pedagogical purpose. Items are discrete units designed to be consumed or interacted with independently or in sequence, see Table C.8.

The Data File represents the digital asset associated with an item, containing the raw or processed content. These files encapsulate the actual instructional material accessed by learners, see Table C.9. Data files are essential for distribution, storage, and computational processing, forming the basis for data extraction, annotation, and analysis within educational platforms.

4.2. ENVIRONMENTAL SET-UP

The following setup is characterized by modularity and containerization[1] (see Figure 4.2) , which facilitates independent development, testing, and deployment of system components, thereby enhancing reproducibility and maintainability. The central coordinating entity is a Python server, implemented with FastAPI[2], which manages orchestration of data flow and interaction among system components.

[1] Containerization is a method of packaging software along with its dependencies and configuration into isolated units called containers. Using tools like Docker, containerization ensures that applications run consistently across different environments by encapsulating everything needed to execute the software. More information is available at https://www.docker.com/.

[2] FastAPI is a modern web framework for building APIs with Python based on standard Python type hints. More information can be found at https://fastapi.tiangolo.com/.

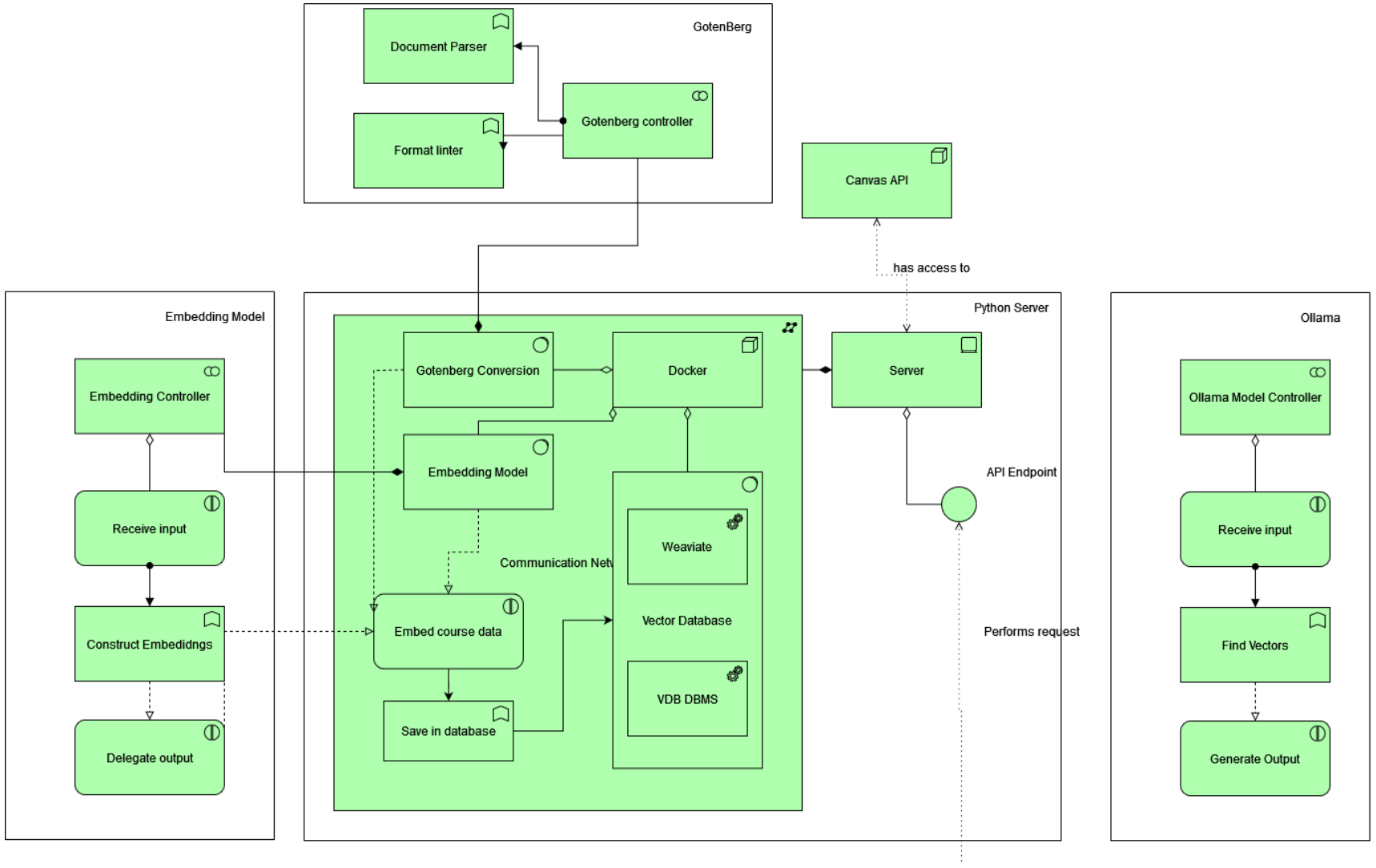


Figure 4.2: TAI - Environemtnal Set-up

4.2.1. DATABASE

The vector database selected for embedding storage and retrieval is Weaviate[3]. This component serves as a pivotal element within the architecture, responsible for storing high-dimensional embeddings that represent the semantic content of educational materials.

Weaviate was chosen for several reasons. Its in-memory storage of vectors facilitates low-latency Approximate Nearest Neighbor (ANN) [78] search, essential for interactive applications requiring real-time retrieval. The database's native support for graph-based indexing structures, such as Hierarchical Navigable Small World (HNSW) [79] graphs, allows for scalable and efficient ANN searches with sub-linear complexity. These graphs organize vectors hierarchically to enable fast search via navigating through layers based on cosine similarity [80].

Additionally, Weaviate provides an extensible schema allowing integration of metadata and hybrid searches combining vector and keyword queries, which enhances retrieval precision[78]. It also offers built-in support for multiple vector embedding models and can accommodate varying embedding dimensions without extensive reconfiguration [79].

Another factor influencing the selection is Weaviate's compatibility with modern container orchestration platforms and cloud-native environments, enabling deployment flexibility and resource isolation. It supports both RESTful APIs[4] and gRPC[5] interfaces for ease of integration with the orchestration server, simplifying system design.

4.2.2. LARGE LANGUAGE MODELS

The generative LLM is deployed using Ollama[6], a system for serving open-parameter language models in local containerized environments. The selected models include Large Language Model Meta AI (LLaMA)[7] 3.1 and 3.2 variants: specifically, the 1B, 3B, and 8B-instruct-q8 models. The 8B-instruct-q8 is a quantized variant[8] of the LLaMA 3.1 architecture. These models offer a range of trade-offs in performance and resource efficiency, with the 1B and 3B models providing lighter-weight alternatives for experimentation and constrained environments.

Ollama was selected due to its compatibility with local deployment strategies and its streamlined support for LLaMA-based models. By avoiding reliance on third-party APIs, the system maintains full control over user data and operational behavior, which is critical in privacy-sensitive educational contexts. Hosting the model locally further allows for reproducibility and

[3] Weaviate is an open-source vector database designed for storing and querying vector embeddings, commonly used in AI and machine learning applications. More information is available at https://weaviate.io/.

[4] https://restfulapi.net/

[5] https://grpc.io/

[6] Ollama is a tool for running LLMs locally with a simple interface and support for various open-source models. It enables fast, offline access to models without relying on cloud services. More information is available at https://ollama.com/.

[7] LLaMA (Large Language Model Meta AI) is a family of large language models developed by Meta AI, designed for efficient and versatile natural language processing tasks. More information is available at https://ai.facebook.com/blog/large-language-model-llama-meta-ai/.

[8] A quantized variant of a machine learning model is a version where numerical precision is reduced to lower precision to reduce model size and improve inference speed, often with minimal impact on accuracy. This is commonly used for deploying models on edge devices or in resource-constrained environments.

transparency.

The model is encapsulated in a container to isolate its computational execution from other system services. This containerization supports modular development, simplifies deployment, and allows targeted performance tuning. Separating the language model from the orchestration logic also enables controlled experimentation with model alternatives, inference strategies, or routing mechanisms. This architecture supports ongoing benchmarking, facilitates system profiling under varying workloads, and allows scalable adjustment of generation capabilities as retrieval or reranking methods evolve.

4.2.3. EMBEDDING MODEL

The system uses a dense text embedding pipeline based on transformer encoders, integrated via the `text2vec-ollama`[9] module in Weaviate. This connects to Ollama, a lightweight runtime serving transformer models for efficient local vectorization.

The embedding model maps natural language sequences into a continuous semantic vector space [81], producing fixed-length vector representations that capture contextual meaning.

Documents are preprocessed and tokenized using the model's vocabulary [82], then passed through the transformer encoder which generates contextualized token embeddings [81]. These token embeddings are pooled (typically by mean pooling) into a single vector [83] which is normalized [84] before being stored in Weaviate's vector index alongside metadata.

Embeddings are computed offline during preprocessing, enabling odular indexing and retrieval that supports dynamic corpus updates and model versioning.

4.2.4. DOCUMENT PROCESSING

To handle the ingestion of heterogeneous educational content, the system incorporates a document processing module capable of converting non-plain-text files into a standardized format. While Portable Document Format (PDF)s are directly parsed within the Python server using text extraction utilities, formats such as Microsoft Word or PowerPoint are not natively parsable due to proprietary structure.

To address this, the system integrates a containerized document conversion service based on Gotenberg[10] (powered by LibreOffice[11] and Chromium[12]), responsible converting office documents to PDF. Gotenberg is selected for its stability, RESTful interface, and compatibility with

[9] https://weaviate.io/developers/weaviate/modules/text2vec-ollama

[10] Gotenberg is an open-source API for converting HTML, Markdown, and Office documents to PDF using headless Chromium and LibreOffice. It is commonly used as a microservice for document generation. More information is available at https://gotenberg.dev/.

[11] LibreOffice is a free and open-source office suite that includes applications for word processing, spreadsheets, presentations, and more. It is widely used as an alternative to Microsoft Office. More information is available at https://www.libreoffice.org/.

[12] Chromium is an open-source web browser project that forms the basis for several browsers, including Google Chrome. It is used for building web-based tools and automations, such as headless browsing for PDF generation. More details can be found at https://www.chromium.org/.

a wide range of office file types. This service enables conversion of unsupported formats into PDFs, after which uniform text extraction can proceed.

By standardizing all documents to PDF prior to parsing, the system ensures consistent preprocessing, which is critical for maintaining the semantic quality of generated embeddings. Encapsulation of the conversion functionality within a discrete container facilitates modular deployment, simplifies maintenance, and supports future extensibility with minimal system-wide impact. This architectural choice improves robustness and allows the retrieval pipeline to operate independently of the source document format.

4.2.5. SEARCH METHODS

Weaviate supports three native retrieval techniques: semantic vector similarity search (using cosine similarity), lexical keyword-based retrieval via BM25, and a hybrid method combining both. These approaches form the basis for our RAG experiments (Table 4.2).

Table 4.2: Supported Search Types in Weaviate

Search Type	Description
Cosine similarity search	Measures semantic closeness between embeddings using cosine similarity [80, 85–88].
BM25 keyword-based retrieval	Uses probabilistic relevance ranking with term frequency and inverse document frequency weighting [85, 89–91].
Hybrid search	Combines normalized semantic and lexical scores to exploit complementary strengths [92, 93].

COSINE SIMILARITY SEARCH

Cosine similarity measures the angle between the query embedding $\mathbf{q}$ and a document embedding $\mathbf{v}_i$ in $\mathbb{R}^d$ space:

$$\text{sim}_{\cos}(\mathbf{q}, \mathbf{v}_i) = \frac{\langle \mathbf{q}, \mathbf{v}_i \rangle}{\|\mathbf{q}\|_2 \|\mathbf{v}_i\|_2} = \frac{\sum_{j=1}^{d} q_j v_{i,j}}{\sqrt{\sum_{j=1}^{d} q_j^2} \sqrt{\sum_{j=1}^{d} v_{i,j}^2}}$$

If embeddings are normalized to unit vectors, cosine similarity reduces to the dot product:

$$\text{sim}_{\cos}(\hat{\mathbf{q}}, \hat{\mathbf{v}}_i) = \hat{\mathbf{q}}^\top \hat{\mathbf{v}}_i$$

This value corresponds to the cosine of the angle θ between $\hat{\mathbf{q}}$ and $\hat{\mathbf{v}}_i$ on the unit hypersphere [86, 88]. The cosine similarity kernel is positive semi-definite [94, 95], which enables its use in kernelized machine learning methods [96].

Efficient ANN search is implemented via HNSW graphs [78, 79, 97], providing logarithmic time complexity in practice [79]. Alternative ANN methods include product quantization [98] and inverted multi-indexes [99]. These approaches balance search accuracy and computational cost [100].

BM25 KEYWORD-BASED RETRIEVAL

BM25 [89] is a widely adopted probabilistic retrieval function that ranks documents by estimating relevance based on Term Frequency (TF) and Inverse Document Frequency (IDF), with document length normalization to mitigate bias towards longer documents [85].

Given a query $Q = \{q_1, \ldots, q_m\}$ and document D_i, BM25 computes:

$$\text{BM25}(Q, D_i) = \sum_{j=1}^{m} \text{IDF}(q_j) \cdot \frac{f(q_j, D_i)(k_1 + 1)}{f(q_j, D_i) + k_1 \left(1 - b + b \frac{|D_i|}{\text{avgdl}}\right)}$$

where $f(q_j, D_i)$ is the frequency of term q_j in D_i, $|D_i|$ is document length, avgdl is average document length, and k_1, b are hyperparameters controlling TF saturation and length normalization [89].

The IDF is computed as:

$$\text{IDF}(q_j) = \log \frac{N - n(q_j) + 0.5}{n(q_j) + 0.5}$$

where N is the total number of documents, and $n(q_j)$ the count of documents containing term q_j, incorporating smoothing [91].

BM25 can also be viewed as a log-linear approximation of the posterior relevance probability in the Binary Independence Model framework [90].

HYBRID SEARCH

Hybrid search combines the semantic similarity from cosine scores and lexical relevance from BM25 by interpolating their normalized values.

For query embedding $\hat{\mathbf{q}}$ and document embedding $\hat{\mathbf{v}}_i$, semantic similarity is:

$$\hat{\text{sim}}_{\cos}(\hat{\mathbf{q}}, \hat{\mathbf{v}}_i)$$

Lexical relevance uses the normalized BM25 score:

$$\hat{\text{score}}_{\text{BM25}}(Q, D_i)$$

Normalization is performed using min-max scaling across the candidate documents:

$$\hat{x}_i = \frac{x_i - \min_k x_k}{\max_k x_k - \min_k x_k}$$

The combined hybrid score is:

$$\text{score}_{\text{hybrid}}(Q, D_i) = \alpha \cdot \hat{\text{sim}}_{\cos}(\hat{\mathbf{q}}, \hat{\mathbf{v}}_i) + (1 - \alpha) \cdot \hat{\text{score}}_{\text{BM25}}(Q, D_i)$$

where $\alpha \in [0, 1]$ balances semantic and lexical contributions [92, 93].

This approach leverages the complementary strengths of semantic generalization and exact lexical matching with minimal latency overhead due to parallel index execution.

4.3. DATA PROCESSING

4.3.1. PREPARATION

The data preparation phase focuses on the assembly and configuration of the retrieval and generation pipeline used in the RAG experiments. This process involves four primary components, including: contextual retriever configuration, prompt construction for language model interaction, session-level chat history management, and integration of retrievers and generation components into callable chains for inference. These components are implemented using the LangChain framework[13], which enables the declarative composition of modular language model workflows. The objective of this phase is to ensure that queries are resolved using relevant context extracted from course materials, while maintaining session coherence.

TEXT EMBEDDING AND VECTORIZATION

Before retrieval and generation can be performed, all educational content must be transformed into a numerical representation suitable for similarity-based search (see Figure 4.3 for full pipeline). This is accomplished through a process of text embedding, wherein raw document segments are encoded into fixed-size dense vectors that capture semantic information.

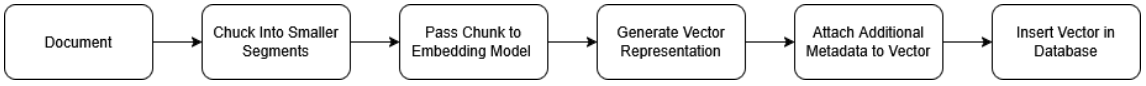


Figure 4.3: TAI - Embedding Pipeline

The input documents, extracted and standardized using LibreOffice (see Section 4.2.4), are segmented into smaller chunks based on a fixed token window size to stay within the embedding model's maximum context length and avoid information loss due to truncation [98, 99]. Metadata such as source filename and resource identifier is attached to each segment to support downstream citation [90].

Each chunk is then embedded using the nomic-embed-text[14] model, integrated via the text2vec-ollama module registered with Weaviate. This transformer-based encoder produces high-dimensional semantic vectors optimized for similarity search, where semantically close inputs yield vectors

[13] LangChain is an open-source framework for developing applications powered by language models. It provides abstractions and utilities for integrating large language models with external data sources, tools, and user interfaces. More information is available at [https://www.langchain.com/.

[14] Nomic Embed Text: a transformer-based embedding model for semantic vector representations. See https://nomic.ai/.

with small Euclidean or cosine distances [81, 83, 84]. The sentence-level embeddings balance inference latency with representational quality, suitable for the vector database interface [86].

The resulting vectors, along with metadata, are stored in the Weaviate vector database to form the searchable index for retrieval [78, 97]. Embeddings are generated offline during preprocessing to decouple ingestion from runtime inference [79].

RETRIEVER CONFIGURATION

Following the embedding of course documents into a vector database, a semantic retriever is instantiated to support contextual information retrieval. This retriever operates over the vector index using similarity-based scoring. It accepts an input query and returns the top-k semantically relevant text chunks based on cosine distance or other ANN metrics provided by the underlying indexing algorithm, see Figure 4.4 for full pipeline.

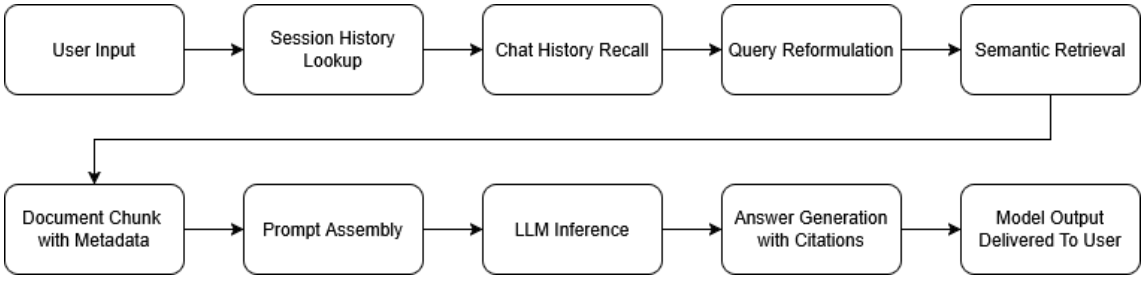


Figure 4.4: TAI - RAG pipeline

To address the challenge of conversational queries referencing prior dialogue, the retriever is augmented with a query reformulation component. This component uses a language model to rewrite the user's latest input into a standalone query by incorporating relevant information from preceding dialogue turns. The reformulated query is then passed to the retriever, ensuring retrieval is conditioned on the full discourse context, even if the user's question is underspecified in isolation.

PROMPT CONSTRUCTION AND OUTPUT STRUCTURING

All language model interactions are governed by structured prompt templates. These templates define a sequence of messages—including system instructions, retrieved document segments, and user queries—that guide the model toward producing task-appropriate outputs, see Table 4.3 and figure C.1.

Table 4.3: Core Components of a Prompt Template

Component	Description
System message	Defines the role and behavioral constraints of the language model (e.g., instructing it to answer questions, classify queries, or cite sources).
Chat history placeholder	Allows injection of prior conversational turns to provide context for current interaction.
User query placeholder	Specifies where the current user input is inserted into the prompt.
Retrieved documents placeholder	Defines the location within the prompt where contextually relevant document segments are inserted to support grounded generation.

In the answering configuration, the output is expected to follow a specific structure. The answer must begin with a concise and instructional explanation that references course content where applicable. This is followed by a structured citation section, presented as a markdown table with two columns: one for the document source (rendered as a clickable hyperlink) and another for the verbatim quote used to support the answer. This format ensures traceability between the generated response and the source material.

SESSION-AWARE MESSAGE MANAGEMENT

To preserve continuity across user interactions, the system maintains chat histories associated with individual session identifiers. Each time a user submits a query, the historical record of all preceding interactions in that session is retrieved and injected into the prompt. This enables the language model to take into account not only the current question but also the conversational path leading to it.

Chat messages are serialized into a role-tagged format, distinguishing between user inputs and model responses. These serialized messages are deserialized and restructured at runtime into the message format expected by the language model interface.

CHAIN ASSEMBLY AND MODULARITY

The final step in the preparation pipeline involves the assembly of retriever, prompt logic, and language model into callable inference chains. Each chain defines a data flow pattern for a specific task, such as question answering or categorization. Components are assembled using a declarative chaining interface provided by the orchestration framework, allowing input-output dependencies to be tracked and controlled pro-grammatically. Table 4.4 depicts the two assembled chains.

Table 4.4: Assembled Chains with Purpose and Output

Chain	Description	Purpose	Output Type
Answering Chain	Accepts user input and session identifier. Retrieves relevant documents, injects history, and generates a structured response with citations.	Generate informative, evidence-backed answers	Structured text with citations
Categorization Chain	Accepts the same inputs but returns a classification label indicating the type or domain of the query, based on retrieved context and internal instruction.	Classify query domain/type for routing or analysis	Classification label

Each chain is configured to operate in both synchronous and asynchronous modes, allowing for immediate response or streaming output where required. In the asynchronous configuration, the system yields partial outputs generated by the model in real time, allowing low-latency user interaction in settings where full answer generation may take several seconds.

MODEL ABSTRACTION AND BACKEND AGNOSTICISM

To allow experimentation with different model architectures, all language model instances are instantiated via a backend-agnostic abstraction. This abstraction selects the appropriate model based on runtime configuration, supporting both local and remote deployment modes. For this

experiment, models were served locally using containerized instances compatible with quantized LLM variants (see Section 4.2.2).

The abstraction also allows models to be swapped or benchmarked without modifying downstream prompt logic or chain structure. This design supports repeatable experimentation and modular benchmarking across varying model sizes, quantization strategies, or inference frameworks.

4.3.2. PROCESSING

The data processing phase operationalizes the components of the retrieval-augmented generation system, applying configuration elements—retriever, prompt templates, history management, and model abstraction—to runtime inference tasks [96, 101]. It transforms static setup into an interactive system that handles real-time user queries and produces structured outputs according to predefined requirements [81, 82].

SESSION RETRIEVAL AND CONTEXT RECONSTRUCTION

Each user query is processed within its session context, identified by a session ID indexing the dialogue history [83, 88]. This history is retrieved, deserialized, and converted into a role-tagged format compatible with the language model interface, ensuring queries are interpreted relative to preceding conversational flow [89]. This supports disambiguation, co-reference resolution, and multi-turn topic tracking [91].

RUNTIME QUERY EXECUTION PIPELINE

Post session reconstruction, queries may undergo context-aware reformulation to create standalone inputs optimized for retrieval [101, 102]. Reformulated queries are passed to the retriever, which performs approximate nearest neighbor search over the embedded corpus using vector similarity [78, 79]. Retrieved segments are organized and injected into prompt templates alongside system instructions and conversation history, composing the final pre-inference payload [92, 103].

INFERENCE AND RESPONSE GENERATION

Prompts are submitted to the inference backend via a model abstraction layer supporting both local and remote deployment [81, 96]. Inference runs in:

- **Synchronous mode:** The model produces a complete output before control returns, suitable for batch processing and logging [97].
- **Streaming mode:** Partial token outputs are emitted incrementally and forwarded to clients in real time, enhancing interactivity and reducing perceived latency [81].

POST-PROCESSING AND OUTPUT STANDARDIZATION

Model outputs are post-processed to meet chain-specific structural and semantic constraints [86]. Answering chains format responses into answer and citation sections with source mappings and excerpts, typically in markdown tables [101]. Categorization chains map outputs to

fixed label sets for routing or analysis [85]. All outputs include metadata such as session ID, model version, inference duration, and token counts to support monitoring, reproducibility, and evaluation [97, 103].

SCALABILITY AND SESSION ISOLATION

The system ensures full session isolation to support concurrent multi-user interaction [93]. Each session maintains an independent chat state stored in memory or external cache, enabling stateless operation, fault tolerance, and horizontal scalability [78]. The processing logic integrates with monitoring tools, load balancers, and autoscaling systems, facilitating production deployments and controlled experiments [79, 97].

4.4. RAG EXPERIMENTAL SET-UP

4.4.1. RAGAS SET-UP

BM25 RETRIEVAL

In the initial RAGAS evaluation framework, BM25 serves as a core component of the hybrid retrieval strategy, providing a robust lexical matching baseline that complements dense vector retrieval methods [78, 85]. BM25 ranks documents based on TF and IDF, while applying document length normalization to mitigate bias towards longer documents [89, 91].

For a query $Q = \{q_1, \ldots, q_m\}$ and document D_i, BM25 computes the relevance score as:

$$\text{BM25}(Q, D_i) = \sum_{j=1}^{m} \text{IDF}(q_j) \cdot \frac{f(q_j, D_i)(k_1 + 1)}{f(q_j, D_i) + k_1 \left(1 - b + b \frac{|D_i|}{\text{avgdl}}\right)}$$

Here, $f(q_j, D_i)$ is the frequency of term q_j in document D_i, $|D_i|$ denotes the document length, and avgdl is the average document length in the corpus. The hyperparameters k_1 and b respectively control TF saturation and document length normalization [89].

Beyond its formula, BM25 is based on the Binary Independence Model (BIM), approximating the log odds of relevance assuming independent binary term features [90]. This probabilistic view explains its balance of term importance and document length.

The parameter k_1 controls term frequency saturation, preventing bias from very frequent terms, while b normalizes document length effects relative to the average, ensuring fair scoring across documents.

COSINE SIMILARITY RETRIEVAL

The evaluation framework employs cosine similarity within a hybrid retriever [78, 79, 97] to rank and extract the top n relevant documents for a given query. Each document and query is encoded into dense vectors [86, 88] enabling efficient cosine similarity computation. Semantic similarity scores are combined with lexical similarity (via ROUGE-L) [100] to evaluate context relevance and guide answer generation.

For the retrieved set $\{d_1, d_2, \ldots, d_n\}$, aggregated context recall is computed as

$$R_c = \frac{1}{n} \sum_{i=1}^{n} \left(\alpha \cdot \text{semSim}(g, d_i) + \beta \cdot \text{lexSim}(g, d_i) \right),$$

measuring coverage relative to the ground truth g. This dual similarity approach helps balance semantic depth and lexical overlap, with weighted constants tuning their influence. These scores are used to assess faithfulness and answer relevancy during evaluation [94–96].

HYBRID RETRIEVAL

The third version of the evaluation framework employs a hybrid retrieval method that separately ranks documents by BM25 and cosine similarity. For a query q, it selects the top k_{bm25} documents by BM25 scores $s_{\text{BM25}}(q, d)$ and the top k_{vector} documents by cosine similarity $s_{\cos}(q, d)$ computed from vector embeddings:

$$s_{\cos}(q, d) = \frac{\mathbf{q} \cdot \mathbf{d}}{\|\mathbf{q}\| \|\mathbf{d}\|}$$

The two result sets, $\mathscr{D}_{\text{BM25}}$ and $\mathscr{D}_{\text{vec}}$, are merged and deduplicated to form the final retrieval set $\mathscr{D}$:

$$\mathscr{D} = \mathscr{D}_{\text{BM25}} \cup \mathscr{D}_{\text{vec}}$$

Unlike the original hybrid scoring function combining normalized BM25 and cosine similarity:

$$\text{score}_{\text{hybrid}}(Q, D_i) = \alpha \cdot \hat{\text{sim}}_{\cos}(\hat{\mathbf{q}}, \hat{\mathbf{v}}_i) + (1 - \alpha) \cdot \hat{\text{score}}_{\text{BM25}}(Q, D_i),$$

this method merges independently ranked results to leverage complementary lexical and semantic strengths without direct score interpolation.

4.4.2. RAGE SET-UP

IF measures the proportion of tokens in the generated answer A that appear in the retrieved context C. Formally, letting

$$T_A = \{t \mid t \in A\}, \quad T_C = \{t \mid t \in C\}$$

the input faithfulness score F_{in} is computed as

$$F_{\text{in}} = \frac{|T_A \cap T_C|}{\max(|T_A|, 1)}$$

OF similarly quantifies the lexical overlap between the answer A and the ground truth G:

$$F_{\text{out}} = \frac{|T_A \cap T_G|}{\max(|T_G|, 1)}$$

where T_G denotes the token set of the ground truth.

SA assesses how well the answer tokens are grounded in the segmented context chunks $\{C_i\}_{i=1}^{n}$. It calculates the fraction of chunks containing any token from the answer, normalized by the total number of chunks:

$$S_{\text{attr}} = \min\left(1, \frac{\sum_{i=1}^{n} \mathbb{1}\left(T_A \cap T_{C_i} \neq \emptyset\right)}{n}\right)$$

where $\mathbb{1}(\cdot)$ is the indicator function evaluating to 1 if the condition holds, 0 otherwise.

For each question-answer pair (q_i, a_i), with retrieved context c_i and ground truth g_i, these metrics are initially computed using lexical overlap heuristics. However, to capture a richer semantic evaluation, a language model is also prompted with these inputs to self-assess the answer's faithfulness and attribution. This process involves formatting a prompt combining the question, concatenated context chunks, generated answer, and ground truth, and invoking the LLM to output scores for input faithfulness, output faithfulness, and source attribution on a continuous scale from 0.0 to 1.0.

The procedural workflow is as follows:

- For each sample, the system formats the prompt by embedding the question, the full retrieved context, the generated answer, and the ground truth.
- The prompt is passed to the LLM, which returns a JSON object containing self-evaluated scores $\hat{F}_{\text{in}}, \hat{F}_{\text{out}}, \hat{S}_{\text{attr}}$.

The collected scores across the dataset form the vectors

$$\mathbf{F}_{\text{in}} = \{F_{\text{in}}^{(i)}\}_{i=1}^{m}, \quad \mathbf{F}_{\text{out}} = \{F_{\text{out}}^{(i)}\}_{i=1}^{m}, \quad \mathbf{S}_{\text{attr}} = \{S_{\text{attr}}^{(i)}\}_{i=1}^{m}$$

for heuristic metrics and

$$\hat{\mathbf{F}}_{\text{in}} = \{\hat{F}_{\text{in}}^{(i)}\}_{i=1}^{m}, \quad \hat{\mathbf{F}}_{\text{out}} = \{\hat{F}_{\text{out}}^{(i)}\}_{i=1}^{m}, \quad \hat{\mathbf{S}}_{\text{attr}} = \{\hat{S}_{\text{attr}}^{(i)}\}_{i=1}^{m}$$

for the LLM self-evaluations.

4.4.3. EVALUATION CONFIGURATION

The evaluation is conducted using a carefully selected set of questions covering a broad range of subjects drawn from diverse educational materials. These materials include summaries, books, and lecture slides (as seen in 4.1).

For each question, relevant documents are retrieved using each retireval approach combining lexical and semantic methods. The retrieved documents, sourced from the aforementioned materials, are concatenated into a single context string. This context, along with the question, is then provided as input to the language model, which generates an answer based solely on the retrieved information without explicitly referencing the source content.

The resulting dataset consists of the original questions, retrieved contexts, generated answers, and corresponding ground truth answers. All data is systematically collected and organized to enable thorough evaluation. Following the data collection, the stored information is forwarded to both RAGAS and RAGE for evaluation.

This entire evaluation process is repeated independently for each retrieval configuration within the retrieval-augmented generation framework. This approach ensures consistent conditions and enables a fair comparison of different retrieval strategies in terms of their impact on answer quality.

By evaluating how different retrieval strategies contribute to accurate, grounded answers, this section ties back to the chapter's opening claim, that TAI in education hinges on transparency and reliability. The methodology outlined here supports this by systematically assessing how well different RAG configurations uphold these principles.

4.5. USER VALIDATION SET-UP

To complement the quantitative evaluation, a qualitative validation was conducted through direct user interaction with the AI. A total of 10 participants, including both students and domain experts, were recruited for this purpose.

Prior to engagement, participants are briefly instructed on the nature and intended functionality of the tool. Following this, each participant is given a 10-minute session to freely interact with the AI, using it to query various educational topics derived from the same materials described in Section 4.1.

Upon completion of the session, participants will take part in unstructured post-usage interviews. These discussions focus on eliciting their perceptions regarding the tool's accuracy, informativeness, and overall trustworthiness. The interviews provide insight into user experiences and perceptions, offering an additional lens through which the system's practical utility and credibility could be assessed.

5

ENTERPRISE ARCHITECTURE

This chapter presents a detailed description of the experimental procedures used to evaluate different EA frameworks and architectures (see sub-sections **??**) within the study. The evaluation focuses creating simulation models of each proposed software architecture, which are then simulated. To assess their effectiveness a wide-range of metrics are used, as seen in sub-section 3.4.3.

5.1. DATA STRUCTURE AND DESCRIPTION

5.1.1. DATA STRUCTURE

To enable the simulation of EA systems in a structured, semantically coherent manner, we adopt ArchiMate[1] as the foundational modeling language. ArchiMate provides a standardized framework for representing the layered structure and dynamic behavior of enterprise systems. Its formalized metamodel captures the core concepts of business, application, and technology domains, along with their interrelations. This makes it particularly suitable for use in simulation, where consistency of structural semantics are essential.

In order to operationalize ArchiMate, we define the underlying data representation as a directed, labeled graph, see Figure D.1.

This graph comprises nodes and edges—nodes represent ArchiMate architectural elements, and edges denote formally defined relationships between them. The graph-based structure mirrors the nature of ArchiMate models, composed of typed, often directional links [104].

It enables efficient data manipulation, traversal, and computation, supporting tasks like model validation and dependency tracking. Graph traversal aligns with enterprise logic flows, en-

[1] ArchiMate is an open and independent modeling language for enterprise architecture, providing instruments to visualize, analyze, and describe the relationships among business domains. More information is available at https://www.opengroup.org/archimate-forum.

abling behavior-driven simulations grounded in structural definitions [105, 106].

Each node stores static properties and dynamic state evolving during simulation. Edges encode semantics and attributes like direction, weight, or operational constraints relevant to simulation [107].

This unified encoding supports both logical architecture and execution. Embedding simulation-relevant properties within nodes and edges facilitates extensibility and integration with graph analysis tools for structure and behavior insights [105, 106].

The node-centric design consolidates relational data in operational units, enhancing modularity and treating nodes as independent, logic-carrying components. It supports diverse interaction patterns and models complex behaviors, where relationships are intrinsic to entities [104, 107].

5.1.2. ABSTRACTION TO VISUALIZATION

The following figure (see Figure 5.1) illustrates the process of data progression within a system, showcasing its journey from initial structuring to final visualization. This process begins with Data Representation, defining the intrinsic structure and inherent properties of the data. Subsequently, this described data progresses to Data and State Management, where mechanisms are employed to control the data's lifecycle, including its storage, retrieval, and reactive updates, thereby maintaining the system's operational state.

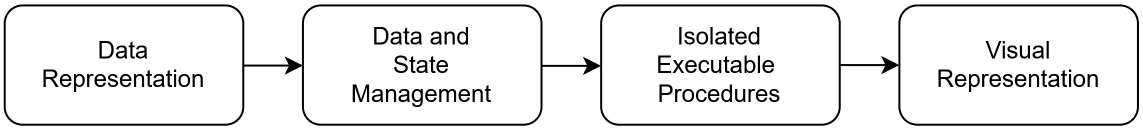


Figure 5.1: Data Abstraction to Visualization Flow

As seen in Figure 5.1, the managed data then transitions to isolated executable procedures. These procedures are meant represent self-contained units of logic designed to operate upon the data. This modular approach enhances system maintainability and reusability. The final stage involves the visual representation of the data. This phase translates the processed data into a perceivable format, enabling comprehensive understanding, facilitating user interaction, and providing an intuitive display of the system's state and operational outcomes.

5.2. ARTEFACT DESIGN

The artefact developed in this project is a web-based application designed to support the modeling and simulation of EA models, directly tying and calling the implemented LLM assistant from Chapter 4. To do so, the application uses the ArchiMate language. The goal of this artefact is to enable the construction of EA models and execute simulations on them via traversing individual and grouped nodes and their associated execution paths, as defined within a modeling environment. For clarity,consistency, and simplicity, the artefact will be referred to as *Flowchitect* throughout the remainder of this document.

The modeling environment allows for the creation of interconnected nodes and relations, reflecting the static and dynamic aspects of the EA models. The Simulation is implemented by traversing these models along execution paths, determined by the structural configuration and flow logic defined by the user. This enables the artefact to serve not only as a design interface but also as an analytical tool for execution-based model validation.

5.2.1. ARTEFACT ARCHITECTURE

The artefact utilizes the Nuxt3[2] framework, which provides a convention-driven directory structure and modular architecture designed to streamline development of web applications, see Figure 5.2.

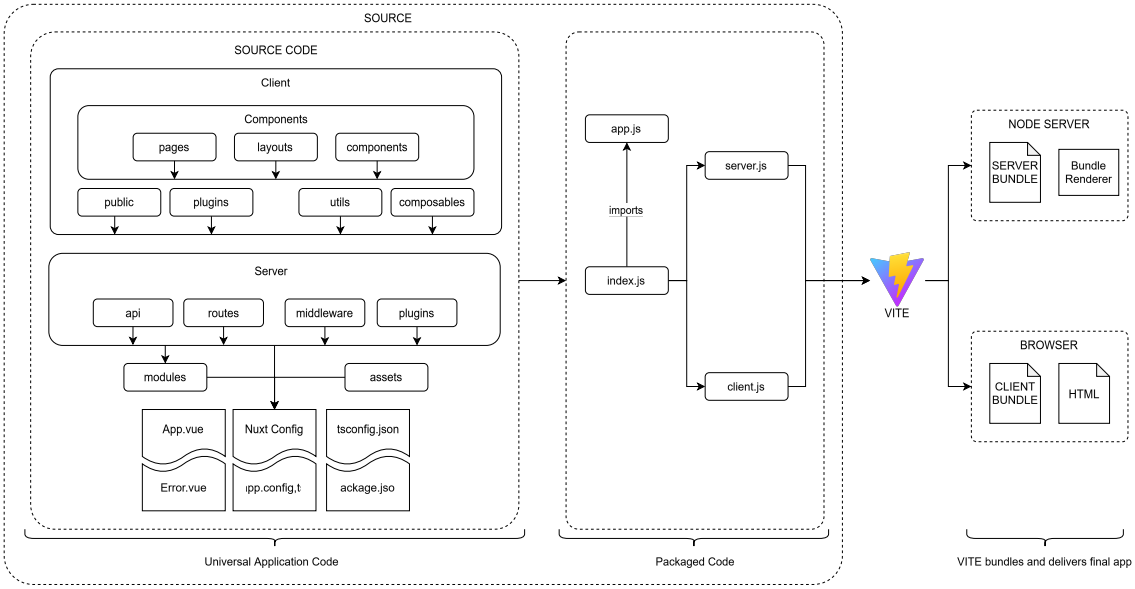


Figure 5.2: Flowchitect - Architecture Format

The initial stage, depicted on the left in Figure 5.2, represents the universal application's source code. This includes client-side elements such as pages, layouts, and reusable components, alongside public assets, plugins, utilities, and composable functions. Concurrently, the server-side structure comprises API endpoints, routes, middleware, server-specific plugins, and modules. Core application configuration files, such as application entry points and build settings, also reside at this level. This holistic organization enables code reuse across both client and server environments. An overview of the source code structure can be found in Table D.1.

The subsequent phase involves packaging this source code into distributable modules, as shown in the middle section of Figure 5.2. This process compiles and transforms the initial source into optimized JavaScript files and associated assets. Finally, Vite[3][4] processes this packaged code,

[2] Nuxt3 is the latest major version of Nuxt, a progressive framework based on Vue.js for building server-rendered or statically generated web applications. More details can be found at `https://nuxt.com/`.

[3] Vite is a build tool that leverages native ES Module imports for development server optimization and bundles code with Rollup for production deployment.

[4] Rollup is a JavaScript module bundler that optimizes code for production via (tree-shaking), resulting in highly optimized and concise bundles.

generating distinct server and client bundles.

Figure 5.3 illustrates a translation process, detailing how the previously defined data structure (see section 5.1) maps to the architectural elements within Flowchitect. The abstract data representation is concretized via schemas, establishing its structure and constraints. Said schemas are defined using Zod[5] to ensure data integrity and validation. The Zod schemas inform the structure and design of dedicated Data Stores, responsible for managing application data and state. Said stores are implemented using Pinia[6], leveraging first class support for reactive, and centralized state management across the application.

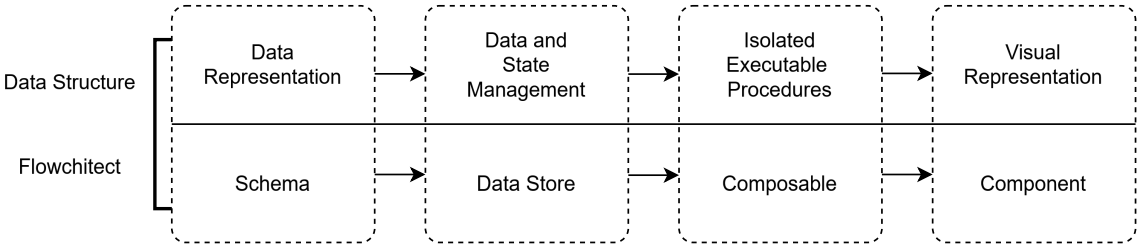


Figure 5.3: Flowchitect - Translation of Data Layer

Further along this translation chain, the abstract concept of isolated executable procedures is realized as Composables[7]. These composables contain logic blocks that operate on the managed data, transforming and processing it accordingly. Said constructs are then utilized and liked with their respective Components, which embody the final visual representation of the data structure.

5.2.2. ARTEFACT PRIMITIVES

NODES

Looking into table D.2 outlines the key behaviors of nodes in the system, which are fundamental for defining how these elements operate and interact within Flowchitect. Each node possesses identity attributes that support clear differentiation and programmatic control. Nodes also support data encapsulation, enabling them to hold operational parameters or contextual information that drive reactive behavior during simulation.

In addition to these core properties, nodes exhibit structural behaviors such as grouping under parent nodes, enforcing boundary constraints, and managing visual stacking through layering. These capabilities facilitate the construction of complex, hierarchical models and help maintain diagrammatic clarity. Connection configurations further define precise attachment points for inbound and outbound links, supporting correct rendering and ensuring logical consistency in graph construction.

[5] Zod is a TypeScript-first schema declaration and validation library, enabling the definition of data schemas and providing parsing and validation functionalities with static type inference. See more at zod.dev.

[6] Pinia is a state management library for Vue.js applications, offering a reactive and type-safe approach to centralizing application state through modular stores. See more at pinia.vuejs.org.

[7] Composable is a functional construct that encapsulates reusable, stateful logic, providing reactive capabilities and promoting modularity across an application. See more at vuejs.org/guide/reusability/composables.html.

Figure 5.4 presents the node lifecycle within the simulation context. It begins with validation of execution eligibility, checking structural and predecessor conditions. Upon activation by the simulation engine, the node verifies its type and properties before executing its assigned logic. Outputs are then generated, stored, and propagated, updating the system state and activating eligible successor nodes.

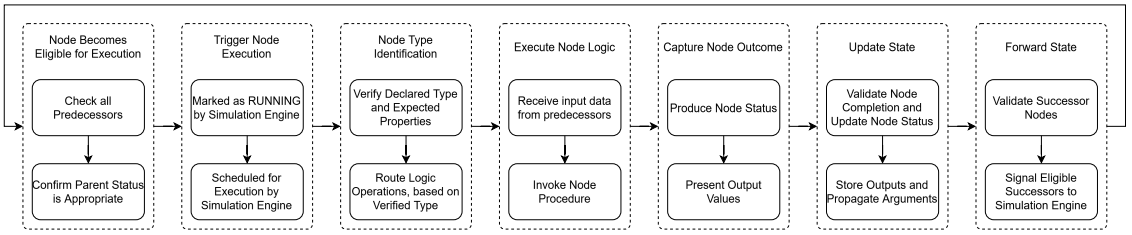


Figure 5.4: Flowchitect - Node Lifecycle

Prior to core logic execution, the node undergoes type identification and property verification, routing to its specific operational procedure. The node then executes its defined procedures, processing inputs to produce an outcome and status. This outcome integrates during a state update phase, where outputs are stored and propagated to downstream components. The lifecycle concludes by forwarding the operational state, validating and signaling eligible successor nodes for subsequent execution.

EDGES

Table D.3 delineates the structural and behavioral features of edges within the system, which are pivotal for encoding directional relationships and interaction logic between nodes. Core identity traits—such as unique identification and explicit source/target associations—form the backbone of edge instantiation, ensuring consistent graph traversal, and precise referencing during runtime execution. The provision for specific endpoint definition further refines this by enabling anchor-level attachment granularity, facilitating complex branching and modular interlinking within dense diagrams.

Table D.3 outlines visual and state-level behaviors that enhance interaction and interpretation. Features like dynamic flow visualization provide immediate feedback on active processes, while updatable connections empower end-users to manipulate graph structure in real-time, supporting exploratory workflows and live system reconfiguration.

5.2.3. ARTEFACT PROCEDURES

This subsection explores the operational logic embedded within the nodes of the platform as composables. Its aim is to detail how individual nodes execute their specific tasks and the mechanisms governing their functional behavior.

In the context of Flowchitect, composables are directly applied to define node behaviors. Each node's operational logic is encapsulated within a distinct composable. The execution of these composables allows each node to perform its designated task, transforming input data accord-

ing to its specific design. Node procedures exhibit two primary operational modalities based on their internal logic.

Figure 5.5: Flowchitect Composables - Direct Computation

The first type performs direct computations (see Figure 5.5). These nodes execute a fixed set of operations on their received inputs, yielding consistent results through deterministic processes. Their operational logic remains invariant, ensuring that identical input states invariably produce identical output states

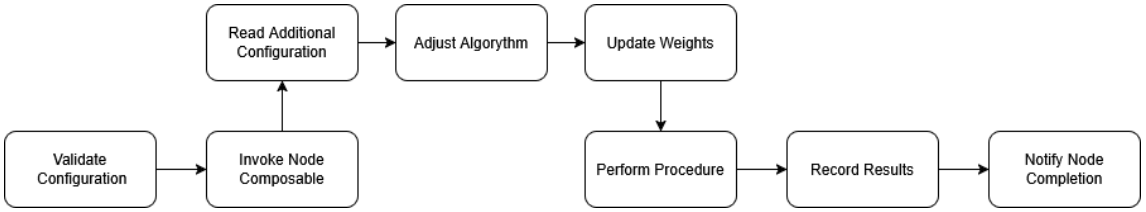


Figure 5.6: Flowchitect Composables - Configurable Computation

The second type incorporates configurable parameters or weighting factors(see Figure 5.6). These elements pre configure the node's internal algorithms, modulating its computational process based on a diverse set of defined configurations or types. This necessity for such diverse configurations arises from the inherent abstraction of certain architectural elements, thereby establishing a structured backbone to perform meaningful computations, which would otherwise be challenging given this high level of abstraction. This allows the node to exhibit varied functional responses from identical input data, as its specific behavior is determined by its initial setup. It is important to note that a node's type or core configuration remains fixed throughout a given simulation run, dictating its operational characteristics for that specific execution. Comprehensive details on all supported types and their respective configurations are provided in tables D.4 to D.12.

5.2.4. ATEFACT SIMULATION LIFECYCLE

The platform's artefact lifecycle embodies a simulation process structured in accordance with MODES-EA. The lifecycle advances through stages including meta-model composition, experiment instantiation, execution scheduling and termination. This structured lifecycle supports the definition, execution, and control of simulation experiments as evolving artifacts, ensuring reproducibility and traceability throughout the experiment's lifespan.

META-MODEL COMPOSITION AND INITIALIZATION

The simulation process is initiated by composing domain-specific and experiment-level metamodels into a unified and formally structured representation (see Figure 5.7). From this composition, an instance graph is generated, comprising interconnected nodes and directed edges

that encode simulation tasks and their respective logical dependencies.

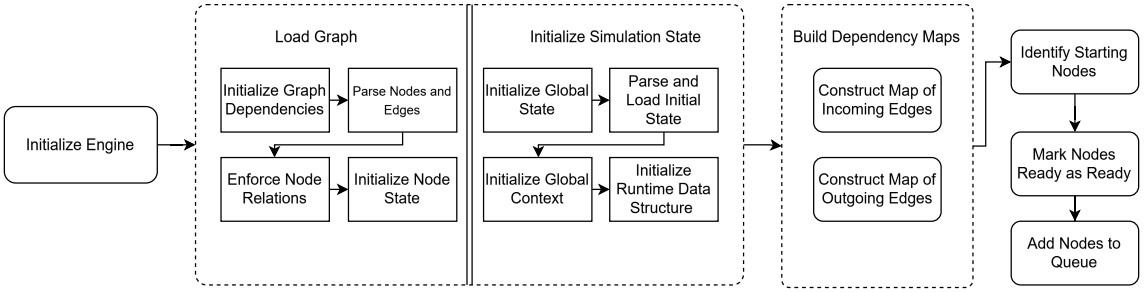


Figure 5.7: Flowchitect - Simulation Initialization

This construction phase aligns with the principles of model-driven simulation engineering as outlined in MODES-EA, where formal meta-models define and constrain the structure of simulation artifacts. The use of explicitly defined model elements establishes a foundation for subsequent execution, ensuring structural integrity. Furthermore, the formalization enables systematic verification and validation by allowing comparison between the modeled behavior and intended experimental objectives, thereby enhancing the credibility and reliability of the simulation setup.

SIMULATION ENGINE

This section details the operational architecture of a simulation engine, a system designed to manage and execute processes defined within a graphical structure. To visually represent figure 5.8 illustrates the fundamental phases of this engine, from its initial setup and graph loading to the iterative execution of defined tasks and the eventual aggregation of results.

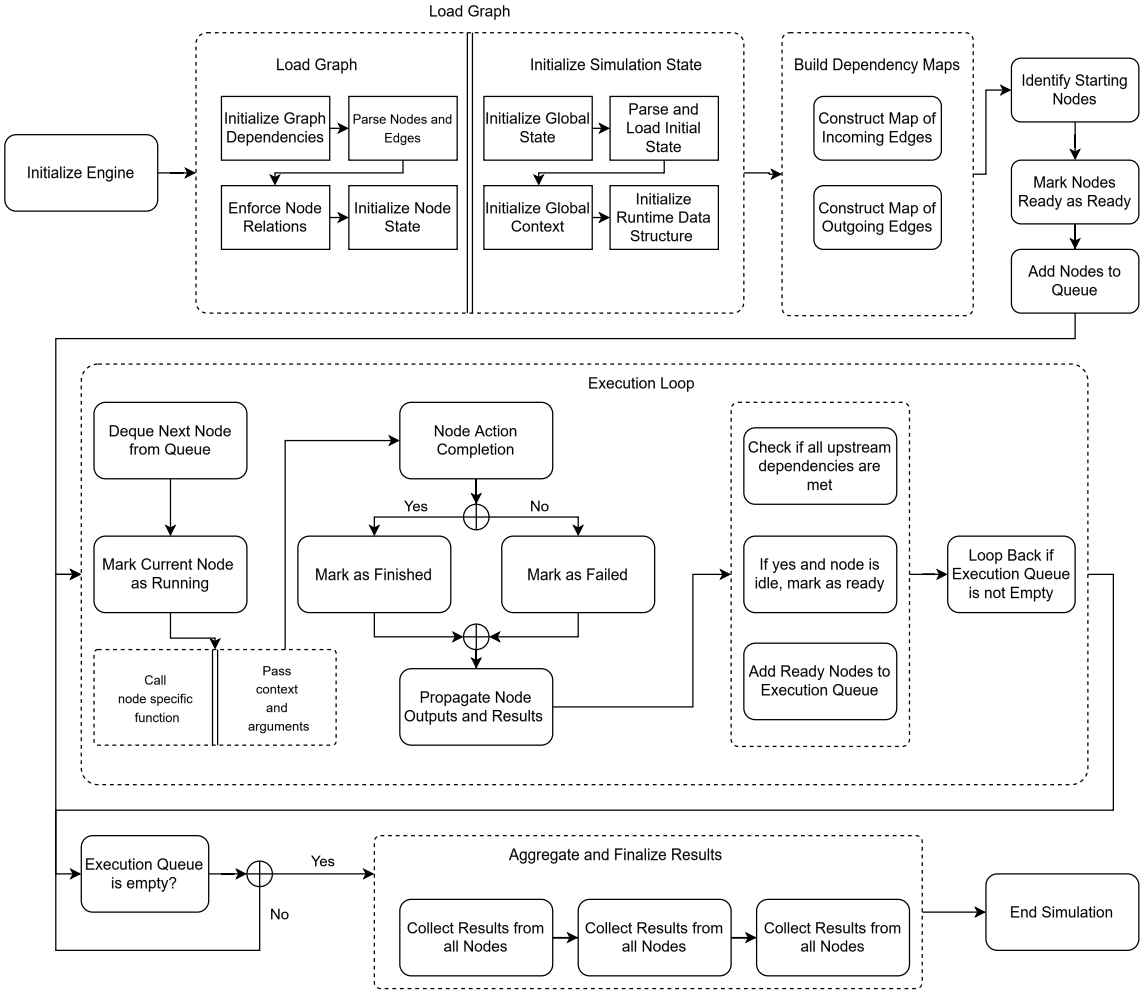


Figure 5.8: Flowchitect - Simulation Engine

The initial phase involves loading the simulation graph, which includes parsing nodes and edges, initializing graph dependencies and node relations, and setting up the engine's initial state. Concurrently, the simulation state is initialized, encompassing global state, context, and runtime data structures. This preparatory stage concludes with building dependency maps for incoming and outgoing edges and identifying starting nodes, which are then marked as ready and added to an execution queue.

Following this, the engine begins its execution loop. Nodes are dequeued, marked as running, and their specific functions are executed. Upon completion, dependencies are checked, and if met, the node's outputs propagate, and it is marked as finished or failed. Ready nodes are then added to the queue. This cycle continues until the queue is empty. Finally, all node results are collected and aggregated, concluding the simulation.

EXECUTION SCHEDULING AND DEPENDENCY CONTROL

The core of the engine's operation is the execution loop, as depicted in Figure 5.9. In this loop, nodes are dequeued, marked as running, and their specific functions are invoked. Upon node action completion, outcomes are determined as either finished or failed, and results are propagated. The system then checks if all upstream dependencies for other nodes are met and proceed add it to the execution queue. This loop continues until the execution queue is empty, signifying the completion of all active processes. Finally, the engine transitions to aggregate and finalize results, collecting data from all nodes before concluding the simulation.

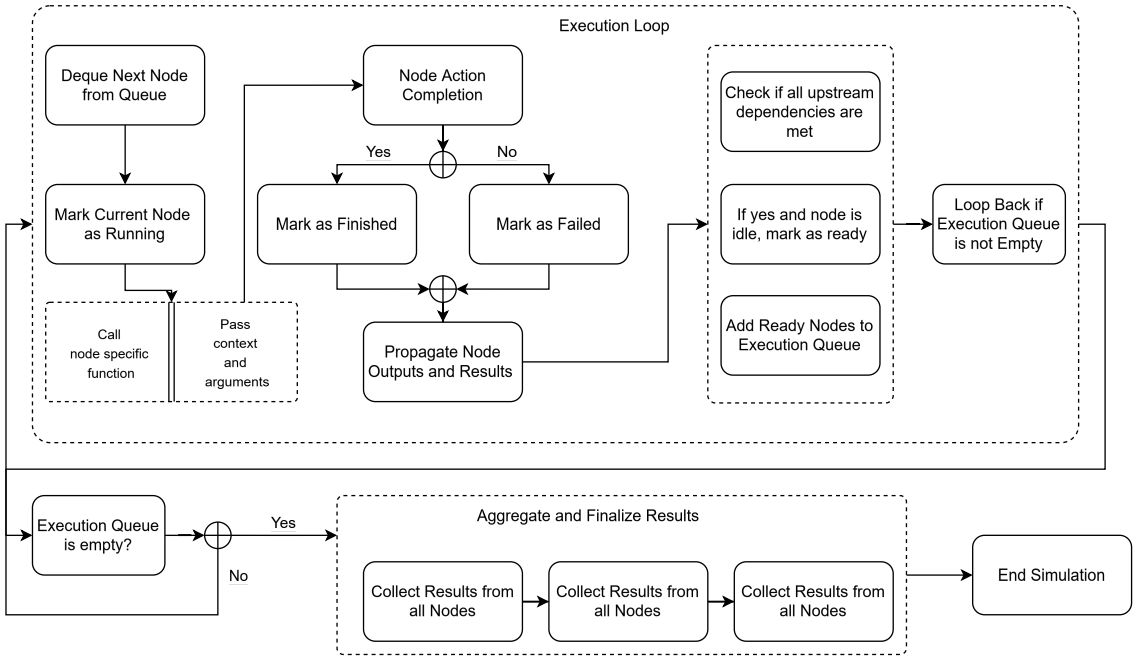


Figure 5.9: Flowchitect - Execution Loop

Execution follows a dependency-driven schedule where each node becomes eligible to run only after all its direct predecessors have completed successfully (see Figure 5.9). This control mechanism enforces causal ordering and data consistency, preventing premature or invalid execu-

tion. The scheduling reflects the experiment's structural dependencies encoded in the meta-model and guarantees that the flow of control and data respects those dependencies throughout the simulation. Reactive state management and asynchronous control flows monitor node statuses and trigger execution only when dependency conditions are met, allowing concurrent execution of independent nodes. The system dynamically computes predecessor and successor relationships via queries on the flow graph, enabling flexible adaptation to structural changes without static control code.

HIERARCHICAL SUB-FLOW HANDLING

The platform supports hierarchical decomposition by organizing nodes into parent-child relationships, forming nested sub-flows, see Algorithm 1. Child nodes execute first, and their completion status governs the parent node's progression. This reflects the concept of compositional meta-models, where experiment structure can be modularized into smaller, manageable units.

Such modular approach aligns with system dynamics and agent-based modeling techniques that emphasize hierarchical and component-based simulation architectures. The execution engine enforces this structure by dynamically resolving containment hierarchies and suspending parent execution until all contained child nodes reach terminal states. Parent status is updated through aggregation of child outcomes, allowing fault propagation or success consolidation as appropriate.

DATA TRANSFER AND EXECUTION BINDING

Nodes collect input data from the output of their immediate predecessors and bind this data to their execution routines, see Algorithm 2. Execution generates output, updates the node's internal state, and makes the results available to successor nodes.

Inputs and outputs are handled explicitly, ensuring transparent data flow, reproducibility, and traceability of transformations throughout the simulation. Each node maintains an independent computational record—including input arguments, execution results, and logs. Data binding and execution triggering are managed reactively, governed by input availability, which preserves consistency and integrity across the flow.

ERROR PROPAGATION AND CANCELLATION

Error states are actively propagated through the execution graph when individual nodes fail during simulation, see Algorithm 3. Upon failure, and if cancellation is configured, all directly dependent successor nodes are recursively invalidated and halted to prevent the continuation of computation based on incomplete or erroneous inputs.

Using systematic flagging and containing error states, the platform ensures that downstream processes do not proceed under invalid assumptions, thereby maintaining the fidelity of simulation outputs and preserving the trustworthiness of experimental conclusions.

TRACEABILITY AND REPRODUCIBILITY

Throughout the simulation lifecycle, node statuses, arguments, computational logs, and output results are persistently maintained and updated within the meta-model instance, see Figure 5.10.

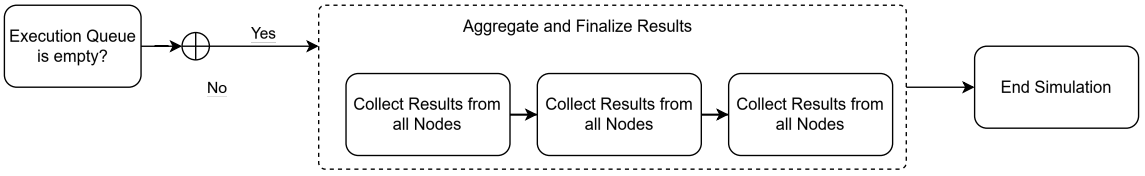


Figure 5.10: Flowchitect - Simulation Results

These updates form a structured execution trace that captures the full progression of the experiment, linking initial inputs to intermediate computations and final outputs. This traceability enables not only reproducibility but also systematic auditing and validation of both the process and its results. Each node encapsulates its own state and artifacts, which facilitates isolated inspection, rollback, and external analysis.

5.3. EA FRAMEWORK SPECIFICATIONS

5.3.1. TOGAF

TOGAF's layered architecture fosters strong strategic-to-operational alignment but struggles with balancing abstraction and practical detail (see Table E.1 for full specifications). While the Strategy Layer excels in defining measurable objectives, its abstraction risks overlooking critical operational realities, potentially causing gaps in execution. The Business and Application Layers provide more granular insights, yet their formalized process models may fail to capture emergent behaviors and dynamic interactions, which are increasingly relevant in agile and cloud-native contexts.

At the Technology and Implementation Layers, quantitative reliability and migration metrics offer concrete progress indicators but are vulnerable to optimistic bias and cost underestimation, which can compromise project outcomes. The framework's modularity and integration with ArchiMate help manage complexity, yet the inherent trade-offs between traceability, detail, and flexibility suggest that TOGAF's practical effectiveness depends heavily on careful tailoring and continuous validation against evolving business and technical landscapes.

5.3.2. ZACHMAN FRAMEWORK

ZF defines a schema for organizing architectural artifacts based on six interrogatives and six stakeholder perspectives [53]. Its taxonomy-based nature implies a need for precise and consistently categorized models. Since it lacks a defined process, the ArchiMate model must impose structure through viewpoint discipline and explicit classification across cells.

The analysis of the ZF across its layers reveals an emphasis on maintaining traceability and separation of concerns, (see Table E.2 for full specifications). For instance, the Strategy Layer

focuses on linking motivation elements to courses of action, ensuring rationale traceability. However, this may introduce bias by overly prioritizing strategic artifacts, potentially constraining operational flexibility. Similarly, the Business Layer demonstrates a modular separation through distinct stakeholder perspectives, but this richness risks fragmentation due to inconsistent granularity across views.

At the Application and Technology Layers, the framework's support for alignment between business functions and application services enhances traceability but may skew toward technical perspectives that overshadow business priorities or fail to capture emergent, dynamic architectures like cloud elasticity. The Technology Layer's explicit hardware-software mapping adds deployment clarity, yet lacks adaptive mechanisms for scaling or evolving infrastructures. Finally, the Implementation and Migration Layer provide structured progress tracking via work packages and migration plateaus, which favors controlled, linear transformations but may inadequately support agile, iterative deployment models.

5.3.3. FEAF

FEAF supports standardized modeling with a focus on performance, governance, and interoperability [54]. ArchiMate modeling must emphasize service standardization, measurement alignment, and cross-agency interoperability.

The FEAF framework exhibits a clear prioritization of federal policy alignment and standardized governance across its layers, evident from qualitative metrics like policy accuracy and governance stability (see Table E.3 for full specification). This focus drives measurable performance outcomes aligned with CPIC and PRM standards but introduces notable biases, particularly a tendency to favor federal-centric goals that may hinder adaptability in more dynamic or non-federal contexts.

At the Business and Application Layers, the framework encourages inter-agency standardization and reuse, with metrics tracking shared services and interoperable interfaces. While this standardization enhances benchmarking and scalability, it also risks stifling innovation by overstandardizing unique agency workflows and prioritizing established interoperability standards. The Technology Layer follows a similar pattern, promoting security and cost efficiency through reusable components, yet this approach can lead to vendor lock-in and reduced openness to emerging technologies.

5.3.4. GARTNER FRAMEWORK

Gartner emphasizes a business outcome–driven approach [55], requiring the ArchiMate model to prioritize strategic alignment and delivery of measurable business value.

The evaluation of the Gartner Framework reveals an emphasis on strategic alignment and outcome-driven architecture. The Strategy layer's focus on aligning architectural outputs with executive goals and tracking value stream throughput aligns well with ArchiMate's capability to model strategic intents as executable initiatives. However, this comes with a notable bias toward

strategic goals that may overshadow technical debt, a limitation partially mitigated by ArchiMate's layered approach. The Business layer prioritizes high-level service abstraction for agile decision-making, which enhances clarity across domains but risks oversimplifying process details. This abstraction suits ArchiMate's service and capability modeling, supporting cross-domain clarity while suggesting a trade-off between granularity and agility.

At the Application and Technology layers, the framework privileges strategic application services and technology abstractions that promote ntegration, aligning with ArchiMate's emphasis on interfaces and service enablement. Yet, the bias toward strategic and high-value applications may neglect legacy systems and infrastructure constraints. The Technology layer's abstraction over low-level details facilitates agility but may obscure operational realities, pointing to the need for complementary detailed infrastructure views in ArchiMate. Lastly, the Implementation and Migration layer's focus on milestone tracking and measurable outcomes supports structured change management but risks underestimating qualitative resistance factors.

5.3.5. DoDAF

DoDAF is a prescriptive architecture framework tailored for defense systems [56]. The ArchiMate model must replicate DoDAF's views through consistent, structured modeling and traceability across complex systems.

The DoDAF framework's layered architecture reflects a strong emphasis on defense-specific requirements, with each layer tailored to support mission-critical and regulated environments. The Strategy and Business layers model mission outcomes, risk, and operational workflows, ensuring alignment with defense objectives and enabling traceability. However, this focus introduces a systemic bias toward defense paradigms, potentially limiting adaptability in commercial or civilian domains, particularly where risk assessments and operational models differ significantly.

At the Application and Technology layers, modularity and interoperability are prioritized, with metrics assessing function coverage and infrastructure compliance to Net-Centric Architecture standards. Yet, these layers also reveal a bias towards defense-grade security and communication protocols, which may hinder integration with broader IT ecosystems that demand more flexibility and agility. This defensive posture is further evident in the Implementation and Migration layer, where adherence to scheduled milestones and compliance gates favors traditional waterfall development cycles, possibly constraining agile methodologies increasingly prevalent in commercial projects.

5.3.6. IAF

IAF provides structured viewpoints across business, systems, information, and technology domains [57].

The IAF's layering reveals distinct emphases that shape architectural rigor and agility. The Strategy Layer prioritizes alignment with long-term goals and scenario completeness, ensuring

strategic coherence across domains, but it risks under-representing emergent tactical needs—a trade-off inherent in structured strategic planning. This tension reflects a classic ArchiMate challenge: balancing high-level intent with adaptability. Similarly, the Business Layer robustly models functions and governance with measurable policy compliance, yet it may miss informal or tacit organizational knowledge, indicating that rigid governance structures could obscure nuanced business dynamics.

The Application and Technology Layers emphasize security, structured flows, and infrastructure sustainability, respectively, with quantitative metrics such as security coverage and integration success ratios. However, both show biases toward conservative approaches—security constraints may stifle innovation, while sustainability metrics might prioritize short-term resource gains. These biases highlight the risk of over-constraining architectural flexibility. Lastly, the Implementation and Migration Layer enforces formal governance checkpoints to mitigate risks, but its procedural correctness could slow agile transitions, suggesting a need for balancing control with adaptability during architectural evolution.

5.3.7. DRAGON1

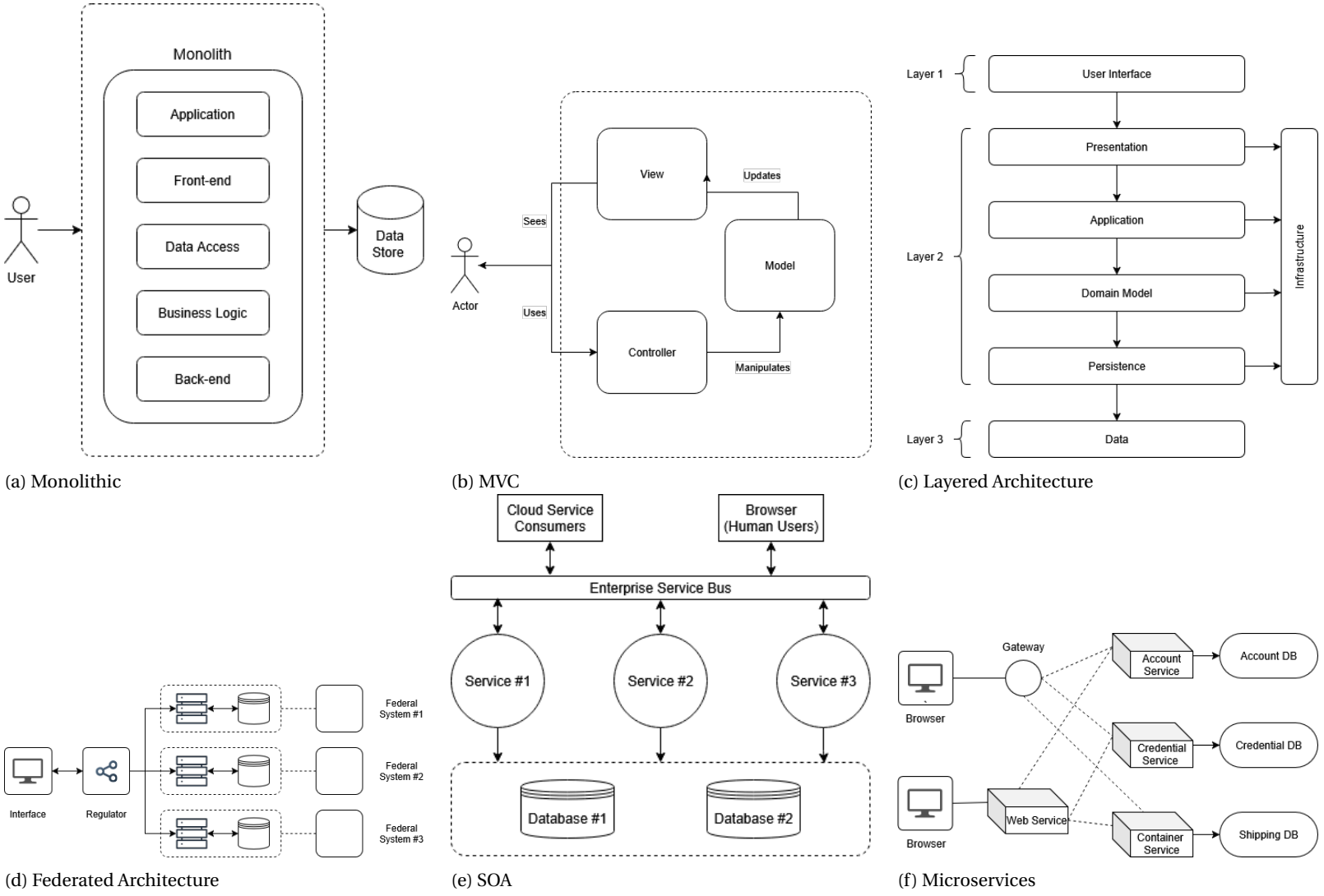
Dragon1 promotes visual enterprise design and stakeholder engagement through clear, transformation-focused visualizations [58].

The framework's layered approach effectively balances qualitative and quantitative metrics to support strategic alignment and stakeholder communication. The Strategy Layer's emphasis on visualizing goals and stakeholder motivations facilitates clear communication, though its abstraction risks oversimplifying complex dependencies. This aligns with ArchiMate's focus on capturing stakeholder intent but highlights a limitation in modeling nuanced strategic interactions, especially in decentralized architectures.

In the Business and Application Layers, the framework prioritizes simplified depictions and tailored views to improve stakeholder engagement and comprehension. While this suits environments with clear service boundaries and modular application landscapes, it may neglect operational detail and marginalize less prioritized stakeholder perspectives. From an ArchiMate viewpoint, this suggests that while the framework supports value stream and service modeling, it requires augmentation to capture dynamic, event-driven business and application interactions more thoroughly.

5.4. PROPOSED SYSTEM ARCHITECTURES

This section presents the system architectures selected for experimental evaluation. The following architectures (see Figure 5.11) were chosen to represent a range of design paradigms, resource configurations, and scalability strategies. Their comparative analysis enables a systematic assessment of performance trade-offs and operational suitability across different deployment contexts. Following this illustration, each architecture is discussed individually to provide a description of their key characteristics, and advantages.



(a) Monolithic

(b) MVC

(c) Layered Architecture

(d) Federated Architecture

(e) SOA

(f) Microservices

Figure 5.11: Employed System Architectures

5.4.1. MONOLITH ARCHITECTURE

Figure 5.11a illustrates the monolithic system architecture. In this design, all application functionalities are encapsulated within a single, unified deployment unit. The architecture is internally organized into several tightly modules, all of which operate within a shared computational environment. External interaction is facilitated through a single entry point, while persistent storage is managed via a centralized data store. This configuration reflects the traditional approach to software design, emphasizing simplicity and cohesion.

As depicted, the monolithic architecture offers advantages due to its centralized structure and reduced component complexity. However, this tightly coupled design also introduces significant drawbacks in terms of scalability, maintainability, and technological flexibility. Any modification within one module can propagate unintended effects across the system, complicating updates and hindering long-term evolution. Scaling the system necessitates replicating the entire application, leading to inefficient resource utilization.

5.4.2. MODEL VIEW CONTROLLER ARCHITECTURE

Figure 5.11b illustrates the Model View Controller (MVC) architectural pattern. This architecture decomposes the application into three distinct components, each responsible for a specific aspect of the system's functionality. The Controller serves as the entry point for user interactions, directing input to the Model, which encapsulates the application's data. Changes in the Model automatically propagate to the View, ensuring the user interface remains synchronized with the underlying data.

As described, the MVC architecture enhances maintainability and modularity by decoupling data handling, presentation, and control logic. Furthermore, by isolating the business logic within the Model and handling input through the Controller, MVC enables dynamic user interfaces that remain consistent with application state, making it particularly well-suited for interactive and scalable software systems.

5.4.3. LAYERED ARCHITECTURE

Figure 5.11c depicts the Layered Architecture pattern, a structural approach that organizes software components into distinct horizontal layers. The architecture comprises six primary layers: User Interface, Presentation, Application, Domain Model, Persistence, and Data. Each layer interacts primarily with its immediate neighbors, promoting a clear separation of concerns and facilitating modular development and maintenance.

The layered architecture offers advantages in terms of maintainability, and modularity. By isolating responsibilities across layers, the architecture supports independent development and modification of components, enabling systematic evolution of the system without affecting unrelated parts. Although not inherently optimized for horizontal scalability, layered systems remain a robust and widely adopted choice in enterprise software, particularly where organizational clarity and long-term maintainability are critical.

5.4.4. SERVICE ORIENTED ARCHITECTURE

Figure 5.11e depicts the Service Oriented Architecture (SOA) pattern, characterized by a collection of loosely coupled, reusable services integrated through an Enterprise Service Bus (ESB). The ESB serves as a middleware layer that mediates communication between service consumers and autonomous services. Each service encapsulates specific business functionality and exposes standardized interfaces, enabling interoperability within the enterprise ecosystem.

SOA facilitates system agility and scalability by promoting service reusability and loose coupling. The ESB centralizes concerns, thus simplifying service integration and management. This architecture supports heterogeneous technologies and allows independent service deployment. By abstracting service interactions through standardized interfaces, SOA enhances enterprise-wide interoperability and enables flexible composition of business processes across diverse systems.

5.4.5. FEDERATED ARCHITECTURE

Figure 5.11d presents the Federated Architecture pattern, a decentralized design approach where autonomous services collaborate through a unified gateway. At the core is the Federated Gateway, which serves as the single point of entry for external requests and routes them to appropriate backend services. Each service encapsulates its own business logic, and dedicated data store, ensuring operational independence and domain-specific data management. This model supports the composition of distributed systems where individual components maintain control while contributing to a cohesive whole.

Federated architecture promotes modularity and resilience by enabling services to function independently yet cooperatively. Decentralized data ownership enhances compliance and autonomy, while well-defined APIs facilitate interoperability across services. By abstracting service orchestration behind the Federated Gateway, the system achieves a balance between autonomy and integration, allowing for efficient, adaptable, and maintainable software ecosystems.

5.4.6. MICROSERVICES ARCHITECTURE

Figure 5.11f illustrates the Microservices Architecture pattern, where the application is decomposed into a collection of small, autonomous services, each responsible for a specific business capability. Clients interact with the system via a centralized API Gateway or Load Balancer, which routes requests to individual microservices. Each service operates independently, owning its own logic and data, enabling flexible deployment and scaling.

The microservices architecture enhances modularity and scalability by enabling independent development and deployment of services. Loose coupling between services reduces interdependencies and allows for diverse technology stacks within the same system. This approach supports rapid iteration, fault isolation, and efficient resource utilization, making it suitable for

complex, evolving applications requiring agility and scalability.

5.5. EA EXPERIMENTAL SET-UP

5.5.1. EA MODEL CONSTRUCTION

MONOLITH EA MODEL

At the application layer (see Figure F.1), the system consists of a single, tightly integrated component within one unit. This unified structure simplifies the simulation of internal interactions, as all processing occurs within the same component. However, the tight coupling means that simulating isolated scaling or failure scenarios for individual functionalities is limited, reflecting real-world constraints on modularity and maintainability.

At the technology layer (see Figure F.2, the entire application runs within one runtime environment on a single server or virtual machine, with an external database for persistent storage. Because the application is tightly packaged as a single entity, the simulation must treat resource consumption and failures at the application level holistically, without the granularity that distributed architectures offer. The simulation highlights how the architecture's simplicity eases deployment and management but constrains flexibility, scalability, and fault isolation.

MODEL VIEW CONTROLLER EA MODEL

The application layer model (see Figure F.3) illustrates how user inputs are processed, triggering data retrieval or updates, which subsequently lead to presentation updates.

In modeling this architecture for simulation, some complexities inherent to MVC are abstracted to focus on primary functional flows. This abstraction supports efficient tracing of interactions and identification of performance bottlenecks without overwhelming the model with excessive detail. The technology layer (see Figure F.4) maps the application components onto typical infrastructure tiers, reflecting a common deployment pattern that separates user interface delivery, business logic processing, and data storage.

The model also highlights trade-offs typical of MVC architectures. While separation of concerns improves flexibility and modularity, it introduces additional communication overhead due to the need for inter-tier coordination.

LAYERED EA MODEL

Although presented in two separate figures (see Figures F.5) and F.6), both views illustrate different perspectives of the same underlying architecture and are thus discussed jointly.

The modeled layered architecture simulation showcases a clear separation of concerns through horizontally organized components (see Figure F.5). The application layer delineates the presentation, business logic, and data access functionalities, while the technology layer specifies their concrete deployment on web servers, application servers, and database servers, respectively (see Figure F.6).

In constructing the model, a balance was struck between detail and simulation tractability. Al-

though real-world layered systems may include additional infrastructure elements and complex service compositions, this model focuses on the fundamental layered interactions. This abstraction facilitates analysis while maintaining fidelity to the core architectural principles.

By adhering to said design strategies, the model remains true to the layered architecture style by enforcing strict layer boundaries and directed communication between adjacent layers, as visually and structurally captured in the two diagrams.

SERVICE ORIENTED ARCHITECTURE EA MODEL

Following the previously provided description of the SOA, the corresponding EA model was constructed in Flowchitect. Although the model is presented in two separate figures (see Figures F.7 and F.8 both views illustrate different perspectives of the same underlying architecture and are thus discussed jointly.

The modelled SOA simulation model shows a modular and service-driven approach, where the application-level functions are realized through decoupled services and components (see Figure F.7, while the technology layer specifies their concrete deployment and operational context F.8. These services are assigned to distinct application components, each responsible for a specific bounded functionality, thereby promoting modularity and service reusability.

In constructing the model, several trade-offs were made to balance model granularity with simulation tractability. While SOA systems often include extensive service registries, discovery mechanisms, and inter-service policies, the model intentionally abstracts these aspects to focus on application-level orchestration and service composition logic. This decision allowed for a cleaner, more analyzable simulation while preserving the core architectural principles.

The model explicitly includes communication paths and deployment mappings, enabling the simulation to account for inter-service latency, network constraints, and potential fault propagation. Resource dependencies were modeled to support experimentation with infrastructure failures, bottleneck identification, and performance profiling.

FEDERATED EA MODEL

Following the earlier architectural overview, the layered model was implemented in Flowchitect and split into two figures for clarity (see Figures F.5 and F.6). Together, these represent the logical separation of concerns and their physical deployment, respectively.

The modeling process emphasized clear separation of responsibilities and strict layering to reflect how each tier communicates only with adjacent layers. Particular attention was given to capturing the flow of requests from the user interface through the business logic to data access, ensuring an accurate representation of control and data dependencies.

Resource allocation and communication paths were explicitly modeled to support analysis of load distribution and potential bottlenecks. This detail facilitates exploration of deployment scenarios and system behavior under different operational conditions.

Overall, the model stays true to the core principles of layered architecture by enforcing strict layering in both logic and deployment, ensuring that each layer's responsibilities and interfaces remain clearly defined and intact throughout the simulation.

MICROSERVICES EA MODEL

Building upon the microservices architecture represented in Figures F.11 and F.12, the models illustrate strict adherence to key architectural principles such as service autonomy and decentralized data ownership.

The explicit representation of dedicated runtimes and databases for each microservice confirms that fault domains are isolated, limiting failure propagation and enhancing system resilience. Moreover, the presence of an API Gateway centralizes external communication without conflating responsibilities for business logic orchestration, preserving service encapsulation.

From a simulation perspective, this architecture allows for modeling resource allocation and failure impact at the granularity of individual services. Consequently, performance evaluation can focus on per-service bottlenecks and inter-service communication overhead.

The network-centric communication model evident in the technology layer necessitates incorporating latency, bandwidth constraints, and potential network faults into any realistic simulation scenario. Additionally, the clear separation between services facilitates a mapping of software components to physical resources, enabling accurate resource utilization.

5.5.2. SIMULATION CONFIGURATION

The experiment is designed to systematically evaluate and compare the performance and behavioral characteristics of all previously defined architectures. Each architecture is modeled and simulated within the context of previously mentioned EA frameworks.

To ensure statistically reliable results, each unique combination of architecture and framework undergoes 150 independent simulation runs. This approach accounts for inherent variability in system behavior due to stochastic elements such as network latency, resource contention, and failure events, which are explicitly modeled within the simulation environment.

The experimental setup therefore includes:

- **Architectures:** 6 distinct architectural paradigms, each embodying unique structural and operational characteristics.
- **Frameworks:** 7 enterprise architecture frameworks, each imposing different governance principles, standards, and modeling conventions.
- **Simulation runs:** 150 repetitions per architecture-framework pair to achieve statistical confidence and to capture performance variability.

This results in a total of $6 \times 7 \times 150 = 6{,}300$ individual simulation runs.

Each framework guides the architectural modeling by specifying constraints, configuration rules, and governance policies that affect deployment, communication patterns, and resource allocation within the simulated environment. Consequently, these frameworks introduce meaningful variations in how the architectures are instantiated and operate under simulation.

The multiple runs per configuration allow for statistical analysis, including mean performance metrics, variance estimation, and confidence interval construction.

The collected data were analyzed to extract a set of evaluation metrics. These metrics quantify various qualitative and quantitative aspects of the simulated architectures, enabling a robust comparative assessment. Table D.13 summarizes the key metrics computed from the simulation results, detailing their respective focus areas and evaluation purposes.

5.6. RESULT GENERATION

In this section, we present the detailed mathematical formulation underlying the generation of the various scores within the composable. The process begins with the extraction and computation of key metrics from the input graph data, followed by architecture inference, normalization of metric values, calculation of factor scores, aggregation into a composite BITAI score, and finally the computation of coverage entropy. Each step is described precisely below.

5.6.1. ARCHITECTURE INFERENCE

Architecture inference is the task of determining the architectural style

$$\text{arch} \in A$$

of a software system from a vector of graph-based metrics:

$$M = \begin{bmatrix} m_1 \\ m_2 \\ \vdots \\ m_k \end{bmatrix} \in \mathbb{R}^k,$$

where each m_i represents a quantitative attribute computed from the system's architecture graph (e.g., node degree distributions, service-to-monolith ratios, coupling metrics).

CONDITION FUNCTIONS AS BOOLEAN MAPPINGS

Each candidate architecture $a \in A$ is associated with a *condition function*

$$C_a : \mathbb{R}^k \to \{0, 1\},$$

where

$$C_a(M) = \begin{cases} 1, & \text{if the system metrics satisfy the structural constraints of } a, \\ 0, & \text{otherwise.} \end{cases}$$

These conditions are expressed as compositions of atomic predicates $p_{a,j}$, themselves defined by inequalities over metrics and thresholds:

$$p_{a,j}(M) := \mathbb{I}\big[m_{i_j} \Diamond_j \tau_j\big],$$

where:

- $\mathbb{I}[\cdot]$ is the indicator function equal to 1 if the predicate is true, else 0,
- m_{i_j} is the i_j-th metric in M,
- $\Diamond_j \in \{>, \geq, <, \leq\}$ is a comparison operator,
- $\tau_j \in \mathbb{R}$ is a threshold,
- j indexes the predicates relevant to architecture a.

Then the full condition is a boolean expression combining these predicates via logical connectives (AND $\wedge$, OR $\vee$, NOT $\neg$):

$$C_a(M) := f_a\big(p_{a,1}(M), p_{a,2}(M), \ldots, p_{a,n_a}(M)\big),$$

where $f_a : \{0,1\}^{n_a} \to \{0,1\}$ encodes the logical structure.

EXAMPLE: MICROSERVICES CONDITION

For instance, the Microservices architecture can be encoded as:

$$C_{\text{Microservices}}(M) = p_{\text{serviceNodeRatio}} \wedge p_{\text{crossLayerEdgeRatio}} \wedge \neg p_{\text{monolithCoupling}},$$

where:

$$p_{\text{serviceNodeRatio}} = \mathbb{I}[m_{\text{serviceNodeRatio}} > \tau_1],$$

$$p_{\text{crossLayerEdgeRatio}} = \mathbb{I}[m_{\text{crossLayerEdgeRatio}} < \tau_2],$$

$$p_{\text{monolithCoupling}} = \mathbb{I}[m_{\text{monolithCoupling}} > \tau_3].$$

These thresholds τ_1, τ_2, τ_3 are domain-calibrated constants.

VECTORIZED CONDITION EVALUATION

Define the predicate vector for architecture a:

$$\mathbf{p}_a(M) = \begin{bmatrix} p_{a,1}(M) \\ p_{a,2}(M) \\ \vdots \\ p_{a,n_a}(M) \end{bmatrix} \in \{0,1\}^{n_a}.$$

The condition function C_a can be represented as:

$$C_a(M) = f_a(\mathbf{p}_a(M)),$$

where f_a may be a conjunction:

$$f_a(\mathbf{p}_a) = \bigwedge_{j=1}^{n_a} p_{a,j},$$

or a more complex boolean formula.

SCORING AND CONFIDENCE LEVELS

Beyond binary classification, we define a *confidence score* or *architecture matching score* $S_a : \mathbb{R}^k \to [0,1]$ that relaxes the boolean conditions into continuous functions:

$$S_a(M) = \prod_{j=1}^{n_a} \sigma\big(\alpha_j(m_{i_j} - \tau_j)\big),$$

where $\sigma(x) = \frac{1}{1+e^{-x}}$ is the logistic sigmoid function, and $\alpha_j > 0$ controls the sharpness of threshold transition.

This smooth approximation allows ranking architectures by likelihood:

$$\text{arch} = \arg\max_{a \in A} S_a(M).$$

The boolean condition relates to the score by thresholding:

$$C_a(M) = \mathbb{I}[S_a(M) \geq \gamma_a],$$

for some confidence threshold γ_a.

FALLBACK AND MULTI-ARCHITECTURE SCENARIOS

Define the feasible architecture set:

$$F(M) := \{a \in A \mid C_a(M) = 1\}.$$

- If $|F(M)| = 1$, inference is unambiguous:

$$\text{arch} = \text{the unique } a \in F(M).$$

- If $|F(M)| > 1$, apply tie-breaking by confidence score:

$$\text{arch} = \arg\max_{a \in F(M)} S_a(M).$$

- If $F(M) = \emptyset$, apply fallback to default architecture:

$$\text{arch} = a_{\text{default}}.$$

COMPOSABILITY AND FACTOR SCORING INTEGRATION

Each factor scoring function F_i is parameterized by the inferred architecture:

$$F_i : \mathbb{R}^k \times A \to \mathbb{R},$$

allowing architecture-specific scoring behaviors:

$$\text{score}_i = F_i(M, \text{arch}).$$

Generally, the scoring vector can be expressed as a composite function:

$$\mathbf{F}(M) = \mathbf{F}\big(M, \text{arch}(M)\big),$$

where $\text{arch}(M)$ denotes the architecture inferred by the above conditions.

This composability ensures the inference step guides context-aware quality evaluation aligned with architectural paradigms.

5.6.2. NORMALIZATION

Each raw metric value v is normalized to a score between 1 and 10 using known minimum and maximum bounds v_{min} and v_{max} as

$$\text{norm}(v; v_{min}, v_{max}) = \max\left(1, \min\left(10, \frac{(v - v_{min})}{(v_{max} - v_{min})} \times 9 + 1\right)\right).$$

This scaling ensures all scores are bounded consistently.

5.6.3. FACTOR SCORING

After inferring the architecture, the next step is to quantify each factor by transforming raw graph-derived metrics into normalized scores that can be directly compared and aggregated. Each factor f is associated with one or more metrics m_f, which represent measurable properties of the software system such as coupling, cohesion, or service granularity.

To normalize these metrics into scores on a fixed scale—typically from 1 to 10—a linear normalization function is applied. Let m_f be the raw value of metric f, and let $v_{\min_f}$ and $v_{\max_f}$ be the empirical minimum and maximum bounds for that metric, chosen based on domain knowledge or observed data distributions. The factor score s_f is defined as:

$$s_f = \text{norm}(m_f; v_{\min_f}, v_{\max_f}) = \max\left(1, \min\left(10, \frac{m_f - v_{\min_f}}{v_{\max_f} - v_{\min_f}} \times 9 + 1\right)\right).$$

This formula ensures the following properties:

- When $m_f \leq v_{\min_f}$, s_f is clamped to 1, representing the lowest score.
- When $m_f \geq v_{\max_f}$, s_f is clamped to 10, representing the highest score.
- For values in between, the score scales linearly from 1 to 10.

In some cases, factors may depend on the inferred architecture arch. For instance, the factor representing executive sponsorship s_{exec} might be modeled as a discrete function conditioned on arch:

$$s_{\text{exec}} = \begin{cases} 6, & \text{if arch} = \text{Monolith}, \\ 8, & \text{otherwise.} \end{cases}$$

This captures the heuristic that different architectures typically exhibit different patterns of governance or sponsorship intensity.

Moreover, factor scoring may incorporate nonlinear transformations, such as logarithmic or exponential scaling, when the distribution of raw metrics is skewed, or when threshold effects must be emphasized. Formally, a generalized factor scoring function may be represented as:

$$s_f = \text{clip}_{[1,10]}\left(g_f(m_f; \theta_f)\right),$$

where g_f is a parametric transformation function with parameters θ_f, and $\text{clip}_{[1,10]}$ restricts the output to the interval $[1, 10]$.

This normalization and scoring step standardizes heterogeneous metric data into a uniform scale, enabling consistent interpretation and further combination of factors.

5.6.4. BITAI SCORE AGGREGATION

The BITAI score provides a composite, holistic evaluation by aggregating the individual factor scores into a single scalar value. This score reflects the overall architectural health, readiness, or maturity based on the weighted contribution of each factor.

Let the set of p factors be indexed as $\{f_1, f_2, \ldots, f_p\}$, each with a corresponding normalized score $s_{f_i} \in [1, 10]$ and an associated weight $w_{f_i} \in \mathbb{R}_{\geq 0}$ representing its relative importance. The weight vector is denoted as:

$$\mathbf{W_F} = (w_{f_1}, w_{f_2}, \ldots, w_{f_p}).$$

The BITAI score S_{BITAI} is computed as the weighted average of the factor scores:

$$S_{\text{BITAI}} = \frac{\sum_{i=1}^{p} w_{f_i} s_{f_i}}{\sum_{i=1}^{p} w_{f_i}} = \frac{\mathbf{W_F} \cdot \mathbf{s}}{\sum_{i=1}^{p} w_{f_i}},$$

where

$$\mathbf{s} = (s_{f_1}, s_{f_2}, \ldots, s_{f_p}).$$

This aggregation method has several desirable properties:

- Factors with higher weights exert greater influence on the final score, enabling prioritization of critical architectural dimensions.
- The normalization by the sum of weights ensures that the BITAI score remains within the factor score bounds (typically between 1 and 10).
- The scalar output provides an interpretable and comparable index for architectural assessment.

In addition, weights w_{f_i} can be calibrated via multiple methods, including:

- Expert elicitation to capture domain knowledge on factor significance.
- Statistical or machine learning optimization to maximize predictive accuracy or correlation with external quality indicators.
- Sensitivity analysis to assess robustness of the BITAI score against variations in factor importance.

More complex aggregation schemes may incorporate nonlinear functions or interaction terms between factors, expressed as:

$$S_{\text{BITAI}} = h\left(\sum_{i=1}^{p} w_{f_i} s_{f_i}, \sum_{i \neq j} w_{f_i, f_j} s_{f_i} s_{f_j}, \ldots\right),$$

where h is a nonlinear function and w_{f_i, f_j} represent pairwise interaction weights. However, the weighted average remains a fundamental and interpretable baseline.

The BITAI score thus serves as a quantitative synthesis of architectural factors, facilitating decision making, benchmarking, and tracking of architectural evolution over time.

5.6.5. COVERAGE ENTROPY

Coverage entropy quantifies the distribution of nodes across architectural layers, providing insight into how evenly or unevenly the system's components are spread. Consider the set of

layers

$$L = \{l_1, l_2, \ldots, l_q\},$$

where q is the total number of identified layers in the architecture.

For each layer $l \in L$, define the count of nodes assigned to that layer as

$$M_l = \left|\{n \in N : \text{layer}(n) = l\}\right|,$$

where N is the set of all nodes in the system graph, and layer(n) returns the layer to which node n belongs.

The total number of nodes in the system is

$$T = \sum_{l \in L} M_l,$$

which satisfies $T = |N|$.

From these counts, we derive the empirical probability distribution over layers:

$$p_l = \frac{M_l}{T},$$

representing the fraction of nodes contained within layer l.

The coverage entropy CE is defined as the Shannon entropy of this probability distribution, capturing the uncertainty or randomness in layer coverage:

$$CE = -\sum_{\substack{l \in L \\ M_l > 0}} p_l \log_2 p_l.$$

This entropy measure has the following properties:

- $CE \geq 0$, with equality if and only if all nodes reside in a single layer (i.e., $p_l = 1$ for some l, and zero otherwise), indicating minimal coverage diversity.
- $CE \leq \log_2 q$, reaching its maximum when nodes are evenly distributed across all q layers, i.e., $p_l = \frac{1}{q}$ for all l.

Intuitively, higher coverage entropy suggests a balanced architectural layering where components are distributed more evenly, promoting modularity and separation of concerns. Lower entropy implies clustering of nodes within fewer layers, which may indicate layering violations or architectural bottlenecks.

For computational stability, terms with $M_l = 0$ are excluded from the summation, as their contribution is zero due to the limit

$$\lim_{p \to 0} p \log p = 0.$$

In some contexts, a normalized coverage entropy CE_{norm} is useful to allow comparison across architectures with differing numbers of layers:

$$CE_{\text{norm}} = \frac{CE}{\log_2 q} \in [0,1].$$

This normalization scales the entropy relative to its theoretical maximum, facilitating interpretation on a standardized scale.

5.7. ESTABLISHING GROUND FOR COMPARISON

The comparison of architectural paradigms in this study is grounded in the use of shared, quantifiable metrics derived from structural properties of the system graph (as seen above). These metrics are architecture-agnostic and capture observable outcomes such as coupling, layering, and modularity. By focusing on measurable outputs rather than internal technological implementations, the approach enables comparisons across diverse architectural styles.

This metric-centric methodology circumvents the need for low-level technological alignment, allowing architectures to be evaluated on a common ground. It avoids the pitfalls of comparing modality-specific implementations, where traditional comparison would fail due to heterogeneity in tooling, programming models, or deployment environments.

Ultimately, the focus lies on the managerial implications and strategic trade-offs introduced by each architecture. Rather than contrasting implementation-specific characteristics, the analysis prioritizes factors that directly impact decision-making and long-term organizational outcomes.

5.8. USER VALIDATION SET-UP

In line with the VVA framework, the validation of Flowchitect involves interviewing experts from the fields of software engineering and EA. Each expert participates in a dedicated 10-minute session of hands-on interaction with the tool, during which they explore its functionality by simulating ArchiMate-based models without any predefined constraints or instructions. This unstructured exploration is designed to elicit authentic, experience-driven insights into the tool's usability and functional alignment with domain-specific modeling practices.

Following the interaction sessions, unstructured post-usage interviews are conducted to collect feedback across the verification, validation, and acceptance dimensions. Participants are invited to reflect on their experience, evaluate whether the tool aligns with established modeling workflows, and comment on the completeness, correctness, and relevance of the tool's outputs. These interviews are meant to capture impressions related to tool adaptability to real-world contexts, and perceived value within professional environments. The gathered qualitative data ensures that Flowchitect is in alignment with expert expectations and practical modeling needs.

6

ALIGNING EXPERIMENTAL APPROACHES

This chapter consolidates the dual experimental tracks explored in the preceding chapters, with the goal of aligning user-centric trustworthiness evaluation and system-level architectural analysis. The chapter begins by recapping the individual objectives and methodological designs of each experiment, highlighting their distinct contributions to the development of TAI systems. It then explores how these approaches complement each other, offering integrated insights across functional, architectural, and governance dimensions. By analyzing trustworthiness as both an evaluative outcome and a structural design constraint, the chapter establishes a cohesive framework that supports the deployment of AI systems that are not only transparent and reliable but also scalable, compliant, and context-aware.

6.1. RECAP OF EXPERIMENTAL OBJECTIVES

This research incorporated two complementary experimental designs, each addressing a critical dimensions in the development of trustworthy AI systems within interactive decision-support environments.

Chapter 4, focused on evaluating the trustworthiness RAG systems. The primary goal of this experiment was to analyze the interpretability and reliability of RAG-generated outputs from the perspective of end-users. By leveraging quantitative frameworks such as RAGAS and RAGE, the study was able to assess output faithfulness, source attribution clarity, and subjective user trustworthiness. Particular attention was paid to the influence of prompt engineering strategies, document parsing complexity, and the overall responsiveness of the RAG system. These factors played a significant role in determining how well the system's outputs could be explained, verified, and adopted in realistic scenarios.

In contrast, Chapter 5 focused on the system-level architecture and organizational implications of integrating these AI-driven assistants. The second experiment utilized a custom-built

simulation engine named Flowchitect. Said tool allowed for the modeling and simulation of various EA frameworks (see Table 2.7) and deployment patterns. The simulation environment made it possible to evaluate how different architectural approaches support the scalable and governed integration of RAG-powered AI assistants. The analysis involved assessing key attributes such as business-IT alignment, traceability, modularity, governance compliance, and long-term scalability. Ultimately, while Chapter 4 examined user-facing aspects of trustworthiness and transparency, Chapter 5 addressed the structural backbone necessary to operationalize such systems.

Although these two experimental streams differed in scope (user-level versus system-level). They are conceptually aligned in their shared objective: to establish a structured foundation for developing context-aware and trust-based AI systems, integrating both robust architectural design and user-centric transparency to support deployment in sensitive and high-stakes domains.

While the integration of both experimental approaches provides a comprehensive overview, each experiment also holds considerable value when considered independently.

Chapter 4 provides a replicable methodology for assessing perceived trustworthiness, output fidelity, and explanation clarity (elements, relevant in AI deployment). Examples in different areas include but do not limit to:

- In healthcare such a methodology can be adapted to evaluate AI-generated diagnostic recommendations, ensuring that patients and medical professionals understand the rationale behind clinical decisions
- In the financial sector, AI-driven investment advice or risk assessments must be accompanied by clear, traceable explanations to satisfy both regulatory compliance and customer trust.
- In the legal domain, where AI is increasingly used for document review or case law analysis, transparent and verifiable explanations, bound by predefined orders, are crucial to uphold due process and institutional accountability.

On the other hand, Chapter 5 offers a toolkit for evaluating the structural, regulatory, and operational implications of deploying AI systems in a given organization, such as:

- The simulation of architectural patterns allows stakeholders to assess how different deployment strategies impact scalability, governance, and modular integration.
- In public sector infrastructures, for instance, such simulations can help determine how best to meet governance and technical requirements while maintaining system responsiveness and user autonomy.
- In large corporations or academic institutions, understanding whether a service-oriented architecture is more conducive to scalable and policy-compliant AI deployments can in-

fluence long-term digital transformation strategies.

Therefore, even when applied in isolation, each experimental track delivers insights that are broadly transferable across sectors concerned with both the functional and ethical dimensions of AI implementation.

6.2. DEEPER ALIGNMENT AND INTEGRATED INSIGHTS

The most significant contribution of this thesis lies in the synergistic alignment of the two experimental approaches. This alignment emerges along three interrelated axes: functional integration, governance and compliance support, and scalable performance optimization. Together, these dimensions form a holistic framework that bridges the gap between AI system design and AI system delivery.

From a functional integration perspective, Chapter 4 shows that the effectiveness of a trustworthy AI assistant depends not just on its model architecture, but on how well its components are orchestrated. These insights require a system design that supports continuous updates and contextual adaptation. Chapter 5 presents the performance evaluation of various architectures, demonstrating their capacity to deploy explainable and trustworthy AI systems. This enables the seamless integration of the XAI pipeline introduced in Chapter 4 into scalable deployment environments, enhancing practical applicability and governance alignment. System architecture also directly affects RAG-based assistant behavior—shaping response accuracy, coherence, and faithfulness—even with identical LLM components. Thus, architecture influences both perceived trustworthiness and performance.

In terms of governance and compliance, Chapter 4 shows that user trustworthiness depends on consistent explanation quality and auditable provenance. Chapter 5 simulates architectures that support such mechanisms. Without structural safeguards, even interpretable models can become governance risks. The findings thereby connect explainable

6.3. TRUST AS BOTH OUTPUT AND DESIGN CONSTRAINT

The intersection of user-level evaluation and architecture-level simulation leads to a higher-order insight, re-framing the concept of trustworthiness in RAG AI systems. Trust must be understood not merely as a measurable output of AI interaction, but also as a foundational design constraint that influences the lifecycle of AI systems. Rather than treating trustworthiness as an end-point outcome, this research positions it as an integrated goal that must shape the design, implementation, and governance of AI systems from their inception.

Trust must be supported by the alignment between infrastructure constraints and policy enforcement mechanisms. AI systems increasingly operate within environments where data governance policies are encoded not only in documents but in the very infrastructure of access and delivery. The integration of these constraints ensures that trustworthiness is not externally applied, but natively embedded within the AI system's operational contract.

Lastly, an often underappreciated aspect of trustworthiness is semantic coherence across user interactions and system responses. In RAG-based systems, inconsistencies in output tone, abstraction level, or explanatory framing can erode user confidence, even when factual accuracy is preserved. This arises when responses are generated from multiple sources or retrieved under dynamically shifting condition. Mitigating this risk requires not only prompt engineering but also architectural provisions for managing discourse context, session persistence, and retrieval harmonization. These mechanisms, while subtle, are critical in reinforcing the perception of a reliable, context-aware assistant. Hence, trustworthiness in AI cannot be solely attributed to what is generated, but must also be evaluated in terms of how consistently and coherently the system behaves across diverse contexts and over time.

7

RESULTS

This section presents the results from both experimental streams, aligned with the core research objectives: (1) enabling users to query and receive trustworthy responses from the assistant, and (2) evaluating system performance across different implementations and architectures. For the TAI experiment, the results were obtained through direct interaction with the assistant, using the RAGAS and RAGE frameworks to assess the performance. These scores offer a comparative view of each method's effectiveness in generating trustworthy and interpretable responses. In parallel, Chapter 5 used the Flowchitect simulation tool to evaluate how different architectural configurations and governance frameworks affect system-level attributes such as traceability, compliance readiness, and scalability. Together, these findings provide an integrated perspective on both the micro-level behavior of AI-driven assistants and the macro-level infrastructure required for their trustworthy deployment.

7.1. TAI RESULTS

7.1.1. RAGAS RESULTS

BM25 SEARCH

The following subsection discusses the results of the implemented BM25 retriever. see Table G.1 for full list of results., the following table (see Table 7.1) presents an overview summary of the results

Table 7.1: Comparison of BM25 RAGAS Scores between LLaMA 3b and 7b

Metric	3b Average	7b Average	Difference (7b - 3b)
Context Recall	0.69	0.75	+0.06
Context Precision	0.69	0.72	+0.03
Faithfulness	0.63	0.64	+0.01
Answer Relevancy	0.37	0.53	+0.16

The smaller model displays reasonably consistent retrieval performance, with moderate success in identifying and utilizing relevant context. While its factual consistency remains relatively stable, its responses often fall short in aligning with the intended user query. This discrepancy is particularly evident in several queries where answer relevancy is noticeably reduced despite adequate context retrieval. The model shows difficulty maintaining performance on more complex or ambiguous queries, suggesting limitations in its ability to integrate retrieved information into coherent, relevant answers.

The larger model demonstrates broader improvements, particularly in its ability to generate more contextually aligned responses. Its retrieval behavior becomes slightly more precise and inclusive, indicating a more effective filtering and selection of relevant information. While its factual accuracy shows only marginal gains, the improved relevancy of its outputs suggests a stronger capacity to synthesize retrieved content into responses that better match user intent. Nonetheless, certain queries continue to pose challenges, reflecting persistent model-level constraints that are not fully resolved by scaling alone.

Comparing the two we can see that increasing model size yields consistent but uneven benefits. Retrieval metrics show moderate enhancement, while factual grounding remains largely unchanged, pointing to persistent issues in maintaining internal consistency. The most substantial improvement is observed in how well the model translates context into relevant answers, emphasizing that interpretive capacity benefits more from scale than retrieval or factual verification. This suggests that model enlargement primarily enhances contextual comprehension rather than retrieval fidelity or fact-based reasoning.

COSINE SIMILARITY SEARCH

The following subsection discusses the results of the implemented BM25 retriever. see Table G.2 for full list of results., the following table (see Table 7.2) presents an overview summary of the results

Table 7.2: Comparison of Cosine Similarity RAGAS Scores between LLaMA 3b and 7b

Metric	3b Average	7b Average	Difference (7b - 3b)
Context Recall	0.70	0.72	+0.02
Context Precision	0.64	0.69	+0.05
Faithfulness	0.61	0.59	-0.02
Answer Relevancy	0.48	0.54	+0.06

The 3b model displays uneven performance, with context recall generally sustained across most queries, while context precision shows greater variability. Faithfulness fluctuates notably, indicating that semantic retrieval does not consistently translate into factually grounded responses. Although some queries yield coherent answers, others demonstrate a gap between retrieved content and generation quality. A subset of queries shows significantly degraded scores, particularly in both faithfulness and relevancy, pointing to occasional retrieval misalignment or interpretive failures.

The 7b model exhibits more stable behavior across queries, especially in retrieval-related dimensions. Context precision improves, suggesting greater selectivity in the use of semantically retrieved segments. However, faithfulness does not increase in tandem and slightly decreases overall, implying that enhanced retrieval alignment does not inherently lead to better factual adherence. Answer relevancy improves across most queries, reflecting a better capacity to generate responses that align semantically with user intent, even if factual grounding remains a challenge in some cases.

Comparing the two we can see that while scaling up leads to better context selection and modest gains in answer relevancy, it does not guarantee improvements in factual consistency. The slight decrease in faithfulness underscores a tradeoff, where enhanced semantic retrieval may introduce loosely aligned but less accurate context. Gains in precision suggest improved noise filtering in retrieval, but these do not necessarily translate into stronger factual grounding, highlighting the limitations of cosine similarity when used in isolation for RAG-based systems.

HYBRID SEARCH

The following subsection discusses the results of the implemented hybrid retriever retriever. see Table G.3 for full list of results., the following table (see Table 7.3) presents an overview summary of the results

Table 7.3: Comparison of Hybrid Search RAGAS Scores between LLaMA 3b and 7b

Metric	3b Average	7b Average	Difference (7b - 3b)
Context Recall	0.71	0.73	+0.02
Context Precision	0.68	0.75	+0.07
Faithfulness	0.59	0.61	+0.03
Answer Relevancy	0.43	0.58	+0.15

The hybrid setup for the 3b model produces relatively consistent retrieval behavior, with high recall and generally stable precision across queries. However, faithfulness remains variable, suggesting that improved retrieval coverage does not consistently lead to grounded responses. Answer relevancy fluctuates more sharply, especially in queries where retrieved content appears misaligned or only loosely connected to the generation target. The drop in performance on specific queries indicates that while hybrid retrieval improves input quality, its benefit is constrained by the model's limited ability to integrate heterogeneous information sources effectively.

For the 7b model, hybrid retrieval leads to stronger and more uniform performance across all dimensions. Precision benefits are particularly notable, reflecting a more refined selection of retrieved content. Faithfulness shows moderate gains, indicating some improvement in grounding, though this remains a weaker dimension overall. Answer relevancy improves across most queries, suggesting that the larger model leverages the blended retrieval signals more effectively than its smaller counterpart. Nevertheless, a few queries continue to expose generation limitations unrelated to retrieval.

Comparing the two we can see that hybrid retrieval yields moderate improvements in context handling and faithfulness when scaling from 3b to 7b. The largest improvement appears in answer relevancy, reinforcing the notion that larger models are more capable of aligning responses with user queries when provided with diverse and complementary retrieval signals. Precision gains further suggest that hybridization contributes positively to filtering noise. However, the relatively modest change in faithfulness indicates that even with more comprehensive input, model-level grounding constraints persist.

7.1.2. RAGE RESULTS

BM25 SEARCH

The following tables present RAGE scores obtained using a BM25 retriever, evaluating both input and output-level factuality as well as source attribution in responses generated by the LLaMA models (see Table G.4 for all results). The following table presents an overview of the differences between the resultso f the two models.

Table 7.4: Comparison of BM25 RAGE Scores between LLaMA 3b and 7b

Metric	3b Average	7b Average	Difference (7b - 3b)
Input Faithfulness	0.58	0.73	+0.15
Output Faithfulness	0.23	0.35	+0.12
Source Attribution	1.00	1.00	-

For the 3b model, input faithfulness shows moderate variation across queries, indicating inconsistent alignment between retrieved content and the input prompt. Output faithfulness is generally low, suggesting limited factual preservation in the model's responses. Despite these issues, source attribution remains consistent across all samples, reflecting reliable citation behavior when references are required. The disparity between input and output scores highlights a potential disconnect between retrieved information and its faithful integration during response generation.

The 7b model demonstrates improved performance in both input and output faithfulness, with higher consistency across queries. The increase in input-level grounding suggests better comprehension and utilization of retrieved content. Output faithfulness also benefits from scaling, though it remains lower relative to input fidelity, indicating that factual degradation still occurs during generation. Source attribution continues to be perfect, implying that both model sizes are equally capable of linking generated text to cited material when necessary.

Comparing the two we can see that the 7b model outperforms the 3b variant on both faithfulness metrics, with the largest gains observed at the input level. This suggests that larger models are better at preserving the factual structure of retrieved evidence during initial processing. However, the more modest improvement in output faithfulness indicates that factual distortion still occurs during generation, regardless of retrieval quality. The unchanged source attribution score reaffirms that citation reliability is unaffected by model scale under this retrieval setup.

COSINE SIMILARITY SEARCH

The following tables display RAGE evaluation results using a cosine similarity retriever, focusing on input/output factual consistency and source attribution across two LLaMA model sizes (see Table G.5 for full results). THe following table creates an overview of the retirever results

Table 7.5: Comparison of Cosine Similarity RAGE Scores between LLaMA 3b and 7b

Metric	3b Average	7b Average	Difference (7b - 3b)
Input Faithfulness	0.36	0.56	+0.20
Output Faithfulness	0.23	0.36	+0.13
Source Attribution	1.00	1.00	-

With cosine-based retrieval, the 3b model struggles to maintain factual alignment, particularly at the output level. Input faithfulness is limited and inconsistent, indicating challenges in leveraging semantically retrieved content effectively. Output faithfulness remains low across most queries, suggesting factual drift during generation. Nevertheless, source attribution is consistently present, indicating that citation behavior is unaffected by the semantic nature of the retrieval mechanism. The performance suggests that the model has difficulty interpreting the less structured evidence returned by cosine similarity search.

The 7b model benefits from scaling, showing marked improvements in both input and output faithfulness. Retrieved information is more consistently integrated into prompts, and factual accuracy in responses increases across the majority of queries. Despite this, a number of samples still reflect degradation during generation, pointing to persistent limitations in output-level grounding. Source attribution remains complete, consistent with prior retrieval methods. The results indicate that the larger model better utilizes semantically relevant input, though it still exhibits fragility in factual transfer to output.

Comparing the two we can see that substantial gains in faithfulness metrics with increased model size. The most pronounced improvement is in input alignment, followed by a smaller but still meaningful gain in output consistency. These trends suggest that the 7b model is better equipped to interpret and generate from semantically retrieved content. However, the persistence of output-level errors highlights a broader issue with factual preservation in generative decoding. The stable source attribution metric indicates that both models maintain consistent citation practices, independent of model scale or retrieval type.

HYBRID SEARCH

The following tables report RAGE scores under a hybrid retrieval setup, combining BM25 and cosine similarity, for evaluating factuality and citation fidelity across LLaMA models (see Table G.6 for full overview). The following table summarizs the differenses between the two tested models.

Table 7.6: Comparison of Hybrid RAGE Scores between LLaMA 3b and 7b

Metric	3b Average	7b Average	Difference (7b - 3b)
Input Faithfulness	0.48	0.72	+0.24
Output Faithfulness	0.23	0.38	+0.15
Source Attribution	1.00	1.00	-

The 3b model shows moderate alignment between the content retrieved from hybrid sources and the input prompt, but the factual integrity deteriorates notably during generation. Input faithfulness remains relatively stable across queries, whereas output faithfulness is consistently low, suggesting challenges in maintaining factual coherence when generating from mixed retrieval signals. Source attribution is complete across the board, indicating that the model consistently anchors its responses to cited content despite deficiencies in factual retention.

The 7b model demonstrates considerable improvements over its smaller counterpart, particularly in its ability to ground both input and output in retrieved content. Input faithfulness increases substantially, reflecting more effective integration of hybrid retrieval. Output faithfulness improves as well, though it continues to lag behind input alignment, pointing to ongoing issues in preserving factual detail through generation. Source attribution remains fully intact, consistent with trends observed across all retrievers and model sizes.

Comparing the two highlights the effectiveness of scaling in improving factual alignment, particularly at the input level. The hybrid setup yields the largest input faithfulness gain among retrieval types evaluated, suggesting that model scale enables better utilization of lexically and semantically combined evidence. Gains in output faithfulness are smaller but meaningful, reinforcing the persistent difficulty of factual preservation during text generation. As with previous evaluations, source attribution remains unaffected by model size or retrieval method.

7.2. ENTERPRISE ARCHITECTURE RESULTS

7.2.1. EA FRAMEWORKS RESULTS

Enterprise architecture frameworks generally focus more on the application layer than on technology infrastructure (see Table 7.7). This is clear in all frameworks with higher application-layer coverage ratios compared to technology-layer ones. This shows a shared priority on connecting business goals to software and services, but it also suggests less attention to the underlying technology, which could affect system reliability and growth. Across frameworks, processing time increases significantly at the highest percentiles, showing challenges in handling complex workloads efficiently.

The frameworks differ in how well they meet their architectural goals. Gartner stands out with the highest Goal Satisfaction Index, showing strong alignment with business outcomes. In contrast, DoDAF has the lowest goal satisfaction, indicating difficulties in meeting objectives under its strict, defense-focused approach. Zachman scores highest on structural integrity (BitAI Score) but faces similar scalability issues as TOGAF. These differences highlight the trade-offs

between clear structure, strict governance, and flexibility in practice.

Table 7.7: Combined Results Across Frameworks

Metric	Mean	91st Percentile	99th Percentile
TOGAF			
BitAI Score	6.42	6.58	6.58
Technology Layer Coverage Ratio	1.70	1.94	1.94
Application Layer Coverage Ratio	2.08	3.22	3.22
Goal Satisfaction Index	0.62	0.75	0.75
Duration (sec)	10.05	12.56	31.62
Zachman			
BitAI Score	6.87	7.01	7.04
Technology Layer Coverage Ratio	1.65	1.88	1.94
Application Layer Coverage Ratio	1.93	2.67	3.22
Goal Satisfaction Index	0.64	0.77	0.77
Duration (sec)	10.77	13.06	31.57
FEAF			
BitAI Score	6.44	6.61	6.61
Technology Layer Coverage Ratio	1.68	1.94	1.94
Application Layer Coverage Ratio	2.05	3.22	3.22
Goal Satisfaction Index	0.63	0.81	0.81
Duration (sec)	10.33	12.86	31.57
Gartner			
BitAI Score	6.62	6.75	6.75
Technology Layer Coverage Ratio	1.70	1.94	1.94
Application Layer Coverage Ratio	2.16	3.22	3.22
Goal Satisfaction Index	0.79	0.99	0.99
Duration (sec)	10.20	13.00	31.57
DoDAF			
BitAI Score	6.49	6.67	6.67
Technology Layer Coverage Ratio	1.69	1.94	1.94
Application Layer Coverage Ratio	2.07	3.22	3.22
Goal Satisfaction Index	0.37	0.43	0.43
Duration (sec)	9.75	12.49	31.75
IAF			
BitAI Score	6.36	6.52	6.52
Technology Layer Coverage Ratio	1.63	1.88	1.94
Application Layer Coverage Ratio	1.86	2.67	3.22
Goal Satisfaction Index	0.42	0.50	0.50
Duration (sec)	10.37	12.84	15.85
Dragon1			
BitAI Score	6.46	6.64	6.64
Technology Layer Coverage Ratio	1.63	1.88	1.94
Application Layer Coverage Ratio	1.87	2.67	3.22
Goal Satisfaction Index	0.65	0.77	0.77
Duration (sec)	9.96	12.50	14.74

Looking at the data, FEAF shows better goal satisfaction at higher percentiles, due to its policy-driven governance, though this comes with increased processing time. TOGAF also focuses heavily on the application layer but struggles to fully meet its goals, as seen in its lower Goal Satisfaction Index. IAF and DoDAF both show lower goal satisfaction scores, suggesting that strict governance and detailed modeling don't always lead to better results, even if their structural scores are solid and processing times are stable.

Dragon1 offers a different approach with moderate scores but better balance, focusing on clear

visuals and stakeholder communication. This helps it achieve reasonable goal satisfaction without heavy processing overhead. Overall, while most frameworks prioritize business and application layers, they could improve by paying more attention to the technology layer for better resilience. The varied goal satisfaction scores show that having detailed models is not enough—frameworks need to balance governance, strategy, and flexibility to work well in practice.

Zachman stands out with the highest BitAI Score, indicating strong structural integrity and organization. However, its goal satisfaction remains moderate, which suggests that even a well-organized framework can struggle to deliver on practical outcomes without stronger guidance on execution and adaptability. Its balanced technology and application coverage ratios show that it attempts to cover both areas fairly but may lack the flexibility needed for dynamic environments.

Gartner, while showing strong goal satisfaction and application coverage, also faces significant increases in processing time under peak conditions. This suggests that frameworks focused on business outcomes and strategic alignment still need to address scalability and efficiency challenges, especially as architectures grow in size and complexity. These results highlight the ongoing challenge for all frameworks: achieving the right balance between comprehensive coverage, clear business alignment, and practical, scalable execution.

7.2.2. ARCHITECTURE RESULTS

Architectural choices significantly impact system behavior under varying loads, revealing key patterns in performance and goal fulfillment 9 see Table 7.8. Monolithic and MVC architectures demonstrate stable performance with manageable processing times, highlighting their efficiency in tightly coupled and moderately modular systems. Their consistent engagement of application layers supports reliable goal satisfaction, making them strong contenders for predictable workloads.

More distributed architectures like Layered, SOA, Federated, and Microservices reveal increasing complexity through higher layer coverage ratios and longer durations. This suggests that while they enable greater modularity and component interaction, they also introduce overheads that can reduce responsiveness. Notably, Layered architecture shows a clear trade-off between extensive application involvement and diminished goal satisfaction, emphasizing the cost of sequential dependencies.

Table 7.8: Combined Results Across Architectures

Metric	Mean	91st Percentile	99th Percentile
Monolith			
BitAI Score	6.46	6.86	6.86
Technology Layer Coverage Ratio	1.41	1.41	1.41
Application Layer Coverage Ratio	1.00	1.00	1.00
Goal Satisfaction Index	0.71	0.99	0.99

Continued on next page

Table 7.8 – *Continued from previous page*

Metric	Mean	91st Percentile	99th Percentile
Duration (sec)	9.85	10.86	14.61
MVC			
BitAI Score	6.45	6.84	6.84
Technology Layer Coverage Ratio	1.65	1.65	1.65
Application Layer Coverage Ratio	1.44	1.44	1.44
Goal Satisfaction Index	0.62	0.88	0.88
Duration (sec)	9.24	10.83	12.95
Layered			
BitAI Score	6.43	6.80	6.80
Technology Layer Coverage Ratio	1.65	1.65	1.65
Application Layer Coverage Ratio	2.67	2.67	2.67
Goal Satisfaction Index	0.54	0.74	0.74
Duration (sec)	12.11	13.75	15.97
SOA			
BitAI Score	6.68	7.01	7.01
Technology Layer Coverage Ratio	1.59	1.59	1.59
Application Layer Coverage Ratio	1.44	1.44	1.44
Goal Satisfaction Index	0.65	0.89	0.89
Duration (sec)	11.25	12.61	18.02
Federated			
BitAI Score	6.46	6.83	6.83
Technology Layer Coverage Ratio	1.88	1.88	1.88
Application Layer Coverage Ratio	2.67	2.67	2.67
Goal Satisfaction Index	0.52	0.69	0.69
Duration (sec)	8.53	11.77	18.36
Microservices			
BitAI Score	6.66	6.75	7.04
Technology Layer Coverage Ratio	1.94	1.94	1.94
Application Layer Coverage Ratio	3.22	3.22	3.22
Goal Satisfaction Index	0.48	0.62	0.62
Duration (sec)	10.06	15.49	18.23

Among distributed models, SOA stands out by maintaining or improving goal satisfaction despite longer processing times, implying effective coordination mechanisms that balance scalability and function. In contrast, Federated and Microservices architectures struggle with goal alignment, likely due to challenges in managing autonomous components and communication delays. This reflects the inherent tension between decentralization and consistent system performance.

The pronounced increase in duration at high percentiles for Microservices signals significant orchestration overhead under stress, highlighting the need for more advanced orchestration despite its modular advantages. Federated systems show similar duration spikes, paired with lower goal achievement, underscoring synchronization difficulties across distributed subsystems. These insights suggest that high modularity demands sophisticated coordination strategies to avoid performance degradation.

Ultimately, the results highlight that architectural complexity and distribution often come with hidden costs affecting system goals and efficiency. Selecting an architecture thus involves balancing modularity and autonomy against processing overhead, with simpler architectures excelling in stability and distributed ones offering flexibility at the expense of responsiveness.

7.3. USER VALIDATION

Participants within the TAI experiment expressed varying degrees of satisfaction. While some found it moderately helpful, others reported a high level of trustworthiness in the chatbot's responses. This was particularly attributed to the system's design features, such as the inclusion of citations, concise explanations appended to each answer. These elements were frequently highlighted as contributing to the perceived transparency and reliability of the tool.

Following interaction with different retrieval setups, there was a tendency to respond more positively to outputs generated using cosine similarity and hybrid approaches. These methods appeared to produce answers that aligned more closely with user expectations regarding relevance and faithfulness, suggesting an implicit preference for semantic-based retrieval.

Overall, both students and domain experts found the information trustworthy, mainly because of the citations and the system's ability to provide accurate and relevant answers. This supports the idea that the tool's design supports users in getting reliable information that meets their needs in a trustworthy manner.

Experts within the EA experiment confirmed that Flowchitect functioned as intended, with simulation behavior accurately reflecting the semantics and logic embedded in the underlying ArchiMate models. This consistency was seen as a strong indicator of technical correctness and model integrity.

Experts from both domains offered detailed observations on the tool's practical utility. Software engineers praised its potential for integrating behavioral and structural analysis within development cycles. Enterprise architects, on the other hand, emphasized the value of Flowchitect for strategic planning and scenario evaluation, highlighting its support for impact analysis and decision-making under uncertainty.

Several experts remarked that the tool closely mirrored the workflows they followed in their day-to-day modeling activities, increasing its perceived relevance and lowering the barrier to adoption in real-world settings. Multiple experts expressed interest in using Flowchitect in ongoing projects, citing its promise to enhance architectural reasoning and model-based system analysis.

Nonetheless, some limitations were identified. Experts noted a number of concerns regarding steep initial learning curves and broader use-cases, particularly for users unfamiliar with simulation-based modeling. These issues however were regarded as addressable and did not significantly undermine the overall favorable assessment. Rather, they were framed as opportunities for future refinement that, if addressed, could broaden the tool's accessibility and impact.

Overall, expert feedback strongly endorsed Flowchitect as a promising and practically valuable tool for modeling and simulating ArchiMate-based architectures. Its ability to combine theoretical soundness with user-centered design was seen as a key strength, reinforcing its potential role in both academic and professional enterprise modeling contexts.

8

DISCUSSION

This chapter synthesizes the insights derived from the empirical evaluation of RAG-based educational assistants and the simulation of EA strategies. The discussion is structured around the two RQs (see ending of Chapter 1), each addressing distinct yet interdependent dimensions of explainability and trustworthiness in AI systems. Together, the results of both RQs show that explainability and trustworthiness must be treated as systemic properties.

8.1. ANSWERING FIRST RESEARCH QUESTION

This section addresses RQ1 (1.2): How can TAI be integrated into RAG-powered LLM educational assistants to enhance transparency and trust?

Findings are based on direct interactions with the assistant, using RAGAS and RAGE metrics to assess retrieval and explanation quality. A key result is the effect of model scale on input faithfulness. Larger models integrate evidence more reliably, while smaller ones often introduce unsupported content. This shows the need for explanation methods that adjust to model capacity and help surface uncertainty in lower-scale models.

Retrieval strategy plays a central role in explanation quality. Lexical methods provide stronger grounding at larger scales due to clearer token overlap. Semantic retrieval increases variability and weakens evidence alignment in smaller models. Hybrid approaches improve faithfulness with larger models but can complicate evidence traceability. These effects call for explanation strategies that clarify how retrieval choices influence output quality and trust.

Each model was tested in isolation. The system was reset between runs to avoid cross-configuration effects. While source attribution remained high, this did not guarantee factual accuracy. Gaps between input faithfulness and factuality show that well-cited answers may still contain flawed reasoning. This highlights the need for explanations that trace sources and also reveal the model's reasoning.

The findings show that explanations must connect retrieval and generation. Retrieval ensures evidence relevance, while generation adds abstraction. A layered explanation approach is needed, one that shows where the evidence came from and how the model used it. Larger models support more interpretive explanations, while smaller models benefit from uncertainty-aware ones. Overall, TAI should adapt explanation design to retrieval type and model scale to support transparency in educational settings.

8.2. ANSWERING SECOND RESEARCH QUESTION

This section addresses RQ2 (1.2): How do different Enterprise Architecture approaches and patterns influence the integration and output of TAI in AI-powered educational assistants?

EA frameworks shape how system components align with educational goals, affecting how explanations are generated and understood. Strong goal alignment makes explanations more relevant to learners and builds trust by linking outputs to clear pedagogical intent.

Layered coverage across technology, application, and business domains enhances traceability. Systems with well-integrated application layers provide more precise and role-specific explanations. In contrast, fragmented architectures reduce transparency and make reasoning paths harder to follow. Patterns like SOA and Microservices support modular explanation design but require careful coordination to maintain clarity across distributed components.

Monolithic and Layered architectures show a trade-off between simplicity and adaptability. Monoliths are efficient but offer limited explanation flexibility. Layered models support multi-level explanations, helping users access both technical and pedagogical insights. Federated systems add privacy and contextualization but increase the difficulty of aggregating consistent and complete explanations due to distributed logic and data handling.

Architectural patterns such as MVC promote separation of concerns, allowing explanations to target specific functional aspects like data handling or user interaction. Microservices and SOA support scalable, modular explanations but require strategies to manage inter-service consistency. Overall, architectural decisions impact how explanations perform across dimensions like clarity, latency, and alignment with educational goals. Effective TAI integration depends on selecting EA patterns that balance these needs while remaining flexible to user and regulatory expectations.

8.3. PRACTICAL IMPLICATIONS

The findings demonstrate that integrating TAI into RAG-powered LLM educational assistants enhances transparency and trustworthiness, crucial for effective learning environments. By tailoring explanation strategies to model scale and retrieval methods, educational systems can provide users with clearer, more reliable insights into AI-driven decisions. EA approaches that emphasize multi-layer integration and modularity further support scalable and maintainable TAI deployment. These frameworks ensure explanations align with pedagogical goals and facilitate role-specific transparency, benefiting educators, learners, and administrators alike. Beyond education, this approach is transferable to other industries—such as healthcare, finance, and legal sectors—where domain-specific goals and compliance require transparent AI explanations. The modular architectures and layered explanation strategies outlined here provide a replicable blueprint for trustworthy AI integration in diverse contexts demanding both interpretability and governance.

8.4. ETHICAL IMPLICATIONS

Transitioning from TAI to ethical AI requires addressing not only transparency but also fairness, accountability, and privacy in educational assistants. While the results highlight the importance of faithful evidence integration and clear reasoning pathways, ethical considerations extend to ensuring AI systems do not propagate bias or misinformation. EA choices, especially federated models, can enhance privacy by localizing data processing, crucial for protecting sensitive student information under regulatory frameworks. However, ethical deployment demands that explanations also expose potential model limitations, uncertainties, and reasoning flaws, enabling users to critically assess AI outputs. Ultimately, ethical AI in education must balance technical transparency with broader social responsibilities—promoting equitable access, respecting user autonomy, and safeguarding data—thereby fostering trustworthiness grounded in both explainability and principled design.

9

CONCLUSION

This chapter summarizes the main findings, contributions, and limitations of the study. By combining user-focused evaluation of RAG-based assistants with architectural simulations, the research highlights how model design, retrieval strategy, and EA jointly influence transparency and trustworthiness. The results offer practical guidance and theoretical insight into building explainable, scalable, and governance-aligned AI systems for educational contexts.

9.1. LESSONS LEARNED

Model scale plays a central role in determining the faithfulness of generated responses in RAG-based educational assistants. Larger models more reliably integrate retrieved evidence, while smaller ones are prone to introducing unsupported content. Retrieval strategies influence this outcome; lexical methods support stronger evidence alignment, while semantic retrieval can reduce factual consistency, especially with smaller models. Despite consistent citation accuracy, source attribution alone is not enough to ensure factual soundness. This points to the need for explanation frameworks that surface both source provenance and model reasoning.

EA decisions significantly shape the integration of TAI into educational systems. Effective frameworks ensure alignment between pedagogical goals and technical implementation, supporting relevant and traceable explanations. Multi-layer coverage improves transparency by allowing explanation paths across business, application, and technology domains. Software architectures determine explanation modularity and coherence. While Microservices and SOA promote flexibility, they introduce complexity. Monolithic and layered models provide more predictable structures, and federated approaches support privacy but complicate aggregation.

Together, these findings suggest that trustowrthiness emerges from the interaction between model behavior, retrieval configuration, and system architecture. From TAI, explanation quality is tied to model capacity and retrieval fidelity. From EA, it depends on traceable work-

flows and modular coordination. Designing explanations as core architectural elements rather than as add-ons allows educational assistants to generate outputs that reflect both institutional structure and user context.

9.2. SCIENTIFIC CONTRIBUTIONS

This study advances the understanding of explainability in RAG-powered educational assistants by showing how model scale and retrieval strategy jointly affect input faithfulness and output factuality. It demonstrates the need for explanations that not only trace source documents but also clarify the model's reasoning process. These insights support more informed retrieval and model selection strategies aligned with transparency goals. The findings also emphasize a shift from output-focused to pipeline-aware explanation design, positioning retrieval transparency as a core component of explainability in AI-mediated educational contexts.

In parallel, the work highlights how EA choices shape the integration of explainability features. It offers practical guidance on selecting EA frameworks and software patterns that balance transparency, scalability, and adaptability. The analysis shows how architectural decomposition affects traceability, granularity, and latency in explanation delivery. This contributes to a broader perspective where explainability is not only algorithmic but also architectural, embedded in how systems are structured and managed across domains.

Taken together, these contributions show that effective TAI depends on the interaction between technical models and system design. TAI reveals how retrieval-model dynamics shape explanation reliability, while EA provides the structural foundation for delivering explanations in a consistent and context-aware way. This integrated view re-conceptualizes explainability as a coordinated system function, shaped by both cognitive and architectural factors, and adaptable to different user needs and educational settings.

9.3. LIMITATIONS

The findings on TAI are limited by the range of models and retrieval strategies evaluated. The analysis focused on specific retrieval configurations, without examining broader architectures or dynamic retrieval methods. The study also did not analyze the model's internal reasoning beyond its grounding in retrieved content, limiting insight into the cognitive aspects of generation. Furthermor , due computational constraints prevented the inclusion of larger models, narrowing the scope of conclusions regarding scalability.

For EA, the analysis used a simulation tool to study different architectural configurations. While this allowed for structured evaluation across multiple patterns, the results are yet to be validated through implementation in real-world systems or artefacts built on top of the simulated configurations. Factors such as user interface design, cognitive effort, and pedagogical context, though relevant to XAI, were not addressed in this work.

10

Future Work

Future research should develop explanation frameworks that capture both the source of retrieved information and the language model's reasoning. Techniques that reveal reasoning paths within LLMs will help close the gap between input faithfulness and output factuality. This may include introspective methods or attention visualizations to show how evidence shapes responses.

Adaptive explainability methods tailored to lexical, semantic, and hybrid retrieval approaches should be explored. Since one explanation style may not fit all, research could test user trust and understanding across explanation formats matched to retrieval types. Customizing explanations could improve clarity while preserving detail, especially for complex semantic links.

Scaling explainability with model size is important. Larger models allow more detailed inferential explanations, while smaller models need methods highlighting uncertainty and reliability. Developing scalable, model-specific explanation protocols will help deploy RAG systems that provide clear and meaningful explanations to diverse users.

Research on XAI in EA should refine integration strategies and how to align explanation services with strategic goals in different educational settings. Investigating interoperability in Microservices and Federated architectures will clarify technical needs for delivering explanations. Data privacy and transparency in decentralized systems require attention.

Longitudinal studies are needed to evaluate the performance of explanation methods and architectures over time in real educational settings. Research should examine how user trust, comprehension, and learning outcomes change with continued use. Studies must investigate the interaction of explainability with different pedagogical models and diverse learner groups. This will help identify context-specific factors affecting the effectiveness of TAI and EA integration and guide the development of scalable and adaptive AI-powered educational assistants.

REFERENCES

[1] P. Wilsdorf, J. Heller, K. Budde, J. Zimmermann, T. Warnke, C. Haubelt, D. Timmermann, U. van Rienen, A. M. Uhrmacher. A model-driven approach for conducting simulation experiments. Applied Sciences 12 (2022) 7977.

[2] H. Khosravi, S. B. Shum, G. Chen, C. Conati, Y.-S. Tsai, J. Kay, S. Knight, R. Martinez-Maldonado, S. Sadiq, D. Gašević. Explainable artificial intelligence in education. Computers and education: artificial intelligence 3 (2022) 100074.

[3] J. Zerilli, U. Bhatt, A. Weller. How transparency modulates trust in artificial intelligence. Patterns 3 (2022).

[4] D. Kaur, S. Uslu, K. J. Rittichier, A. Durresi. Trustworthy artificial intelligence: a review. ACM computing surveys (CSUR) 55 (2022) 1–38.

[5] W. Holmes, K. Porayska-Pomsta, K. Holstein, E. Sutherland, T. Baker, S. B. Shum, O. C. Santos, M. T. Rodrigo, M. Cukurova, I. I. Bittencourt, et al. Ethics of ai in education: Towards a community-wide framework. International Journal of Artificial Intelligence in Education (2022) 1–23.

[6] X. Duan, B. Pei, G. A. Ambrose, A. Hershkovitz, Y. Cheng, C. Wang. Towards transparent and trustworthy prediction of student learning achievement by including instructors as co-designers: A case study. Education and Information Technologies 29 (2024) 3075–3096.

[7] V. Vera, J. Fu, M. J. Irvin, in: International Conference on Human-Computer Interaction, Springer, pp. 131–137.

[8] M. Saarela, V. Podgorelec. Recent applications of explainable ai (xai): A systematic literature review. Applied Sciences 14 (2024) 8884.

[9] J. Rorseth, P. Godfrey, L. Golab, D. Srivastava, J. Szlichta, in: 2024 IEEE 40th International Conference on Data Engineering (ICDE), IEEE, pp. 5669–5670.

[10] D. Hennekeuser, D. D. Vaziri, D. Golchinfar, D. Schreiber, G. Stevens. Enlarged education–exploring the use of generative ai to support lecturing in higher education. International Journal of Artificial Intelligence in Education (2024) 1–33.

[11] K. Fiok, F. V. Farahani, W. Karwowski, T. Ahram. Explainable artificial intelligence for education and training. The Journal of Defense Modeling and Simulation 19 (2022) 133–144.

[12] D. Wang, C. Bian, G. Chen. Using explainable ai to unravel classroom dialogue analysis: Effects of explanations on teachers' trust, technology acceptance and cognitive load. British Journal of Educational Technology 55 (2024) 2530–2556.

[13] V. Swamy, B. Radmehr, N. Krco, M. Marras, T. Käser. Evaluating the explainers: black-box explainable machine learning for student success prediction in moocs. arXiv preprint arXiv:2207.00551 (2022).

[14] K. Z. Gajos, L. Mamykina, in: Proceedings of the 27th International Conference on Intelligent User Interfaces, pp. 794–806.

[15] D. Hooshyar, Y. Yang. Problems with shap and lime in interpretable ai for education: A comparative study of post-hoc explanations and neural-symbolic rule extraction. IEEE Access (2024).

[16] S. J. Yang, H. Ogata, T. Matsui, N.-S. Chen. Human-centered artificial intelligence in education: Seeing the invisible through the visible. Computers and Education: Artificial Intelligence 2 (2021) 100008.

[17] H. Ogata, B. Flanagan, K. Takami, Y. Dai, R. Nakamoto, K. Takii. Exait: Educational explainable artificial intelligent tools for personalized learning. Research and Practice in Technology Enhanced Learning 19 (2024).

[18] N. Kamawar-MacLeod, D. Thue, in: Proceedings of the 12th International Conference on Human-Agent Interaction, pp. 329–331.

[19] I. Tiddi, S. Schlobach. Knowledge graphs as tools for explainable machine learning: A survey. Artificial Intelligence 302 (2022) 103627.

[20] Y.-L. Chou, C. Moreira, P. Bruza, C. Ouyang, J. Jorge. Counterfactuals and causability in explainable artificial intelligence: Theory, algorithms, and applications. Information Fusion 81 (2022) 59–83.

[21] E. Cambria, L. Malandri, F. Mercorio, M. Mezzanzanica, N. Nobani. A survey on xai and natural language explanations. Information Processing & Management 60 (2023) 103111.

[22] A. Papenmeier, D. Kern, G. Englebienne, C. Seifert. It's complicated: The relationship between user trust, model accuracy and explanations in ai. ACM Transactions on Computer-Human Interaction (TOCHI) 29 (2022) 1–33.

[23] G. Kostopoulos, G. Davrazos, S. Kotsiantis. Explainable artificial intelligence-based decision support systems: A recent review. Electronics 13 (2024) 2842.

[24] J. M. Alonso, G. Casalino, in: International workshop on higher education learning methodologies and technologies online, Springer, pp. 125–138.

[25] Q. Liu, J. D. Pinto, L. Paquette, in: Trust and Inclusion in AI-Mediated Education: Where Human Learning Meets Learning Machines, Springer, 2024, pp. 93–109.

[26] G. Casalino, G. Castellano, P. Ducange, M. Fazzolari, R. Pecori, G. Zaza, in: International Conference on Higher Education Learning Methodologies and Technologies Online, Springer, pp. 95–111.

[27] E. Kasneci, K. Seßler, S. Küchemann, M. Bannert, D. Dementieva, F. Fischer, U. Gasser, G. Groh, S. Günnemann, E. Hüllermeier, et al. Chatgpt for good? on opportunities and challenges of large language models for education. Learning and individual differences 103 (2023) 102274.

[28] A. Renz, G. Vladova. Reinvigorating the discourse on human-centered artificial intelligence in educational technologies. Technology Innovation Management Review 11 (2021).

[29] M. Bhat, D. Long, in: Proceedings of the 2024 ACM Designing Interactive Systems Conference, pp. 939–957.

[30] B. A. Chaushi, B. Selimi, A. Chaushi, M. Apostolova, in: World Conference on Explainable Artificial Intelligence, Springer, pp. 48–71.

[31] K. Jafarzade, in: 2023 5th international conference on problems of cybernetics and informatics (PCI), IEEE, pp. 1–3.

[32] V. Uglev, in: International Conference on Intelligent Tutoring Systems, Springer, pp. 371–380.

[33] I. Ilin, A. Levina, A. Borremans, S. Kalyazina, in: Energy management of municipal transportation facilities and transport, Springer, 2019, pp. 124–142.

[34] J. D. Rittelmeyer, K. Sandkuhl, in: 2021 IEEE 25th International Enterprise Distributed Object Computing Workshop (EDOCW), IEEE, pp. 130–137.

[35] J. Malott. Enterprise architecture for ai-powered digital transformation (2024).

[36] N. E. Kamalabai, I. Donoghue, L. Hannola, in: 2024 Portland International Conference on Management of Engineering and Technology (PICMET), IEEE, pp. 1–12.

[37] L. Fitriani, M. L. Khodra, K. Surendro, in: 2023 International Conference on Computer, Control, Informatics and its Applications (IC3INA), IEEE, pp. 90–95.

[38] J. Sayles, in: Principles of AI Governance and Model Risk Management: Master the Techniques for Ethical and Transparent AI Systems, Springer, 2024, pp. 373–382.

[39] R. Dilnutt, D. Xiao, G. Fang, J. Yang, R. Ran, L. Wu, How does enterprise architecture support artificial intelligence-involved digital transformation in the manufacturing industry, 2023.

[40] A. Zimmermann, R. Schmidt, D. Jugel, M. Möhring, in: Research Challenges in Information Science: 14th International Conference, RCIS 2020, Limassol, Cyprus, September 23–25, 2020, Proceedings 14, Springer, pp. 145–153.

[41] D. Denni-Fiberesima, in: International Conference on Business and Technology, Springer, pp. 210–222.

[42] L. Wang, J. Zhao, Strategic Blueprint for Enterprise Analytics, Springer, 2020.

[43] K. Sandkuhl, J. D. Rittelmeyer, in: Enterprise Engineering Working Conference, Springer, pp. 149–166.

[44] Y. Martel, A. Roßmann, E. Sultanow, O. Weiß, M. Wissel, F. Pelzel, M. Seßler, in: INFORMATIK 2020, Gesellschaft für Informatik, Bonn, pp. 165–181.

[45] H. Takeuchi, S. Yamamoto. Ai service system development using enterprise architecture modeling. Procedia Computer Science 159 (2019) 923–932.

[46] S. Lolo, E. Kaburuan, N. Legowo. Analysis of enterprise architecture using the togaf framework in educational services. Int. J. Adv. Sci. Technol 29 (2020) 3386–3400.

[47] A. Abdallah, A. Abran, M. A. Khasawneh. Enterprise architecture measurement: A systematic literature review. Journal of Theoretical and Applied Information Technology 99 (2021) 1257–1268.

[48] M. Missikoff. A knowledge-driven business process analysis canvas. arXiv preprint arXiv:2201.06860 (2022).

[49] D. O'Higgins. Impacts of business architecture in the context of digital transformation: An empirical study using pls-sem approach. arXiv preprint arXiv:2307.11895 (2023).

[50] L. Urbaczewski, S. Mrdalj. A comparison of enterprise architecture frameworks. Issues in information systems 7 (2006) 18–23.

[51] A. Singh, P. Mudholkar, in: ICAIM-International Conference on Advancement in IT and Management.

[52] A. Josey, TOGAF® version 9.1-A pocket guide, Van Haren, 2016.

[53] J. A. Zachman. A framework for information systems architecture. IBM systems journal 26 (1987) 276–292.

[54] B. Bellman, F. Rausch, in: International Conference on Electronic Government, Springer, pp. 48–56.

[55] R. S. Bittler, G. Kreizmann. Gartner enterprise architecture process. Evolution 21 (2005).

[56] U.S. Department of Defense, Department of Defense Architecture Framework (DoDAF), U.S. Department of Defense, 2003. URL: https://dodcio.defense.gov/Library/DoD-Architecture-Framework/, version 1.0.

[57] J. Van't Wout, M. Waage, H. Hartman, M. Stahlecker, A. Hofman, The integrated architecture framework explained: why, what, how, Springer Science & Business Media, 2010.

[58] D. A. Foundation, Dragon1: Enterprise architecture framework for visual enterprise design, https://www.dragon1.com/resources/enterprise-architecture-framework, 2025. URL: https://www.dragon1.com/resources/enterprise-architecture-framework, dragon1 is a visual enterprise architecture method and open EA framework developed to support enterprise innovation and transformation. It emphasizes visual modeling and stakeholder engagement in architectural practices. Accessed: 2025-05-20.

[59] S. Kotusev. A comparison of the top four enterprise architecture frameworks. British Computer Society (BCS) 1 (2021) 1–10.

[60] B. D. Rouhani, M. N. Mahrin, F. Nikpay, P. Nikfard, in: 2013 international conference on informatics and creative multimedia, IEEE, pp. 1–6.

[61] A. Dresch, D. P. Lacerda, J. A. V. Antunes Jr, A. Dresch, D. P. Lacerda, J. A. V. Antunes, Design science research, Springer, 2015.

[62] G. S. Fishman. Principles of discrete event simulation.[book review] (1978).

[63] J. Banks, J. S. CARSON II, L. Barry, et al., Discrete-event system simulationfourth edition, 2005.

[64] J. A. Sokolowski, C. M. Banks, Principles of modeling and simulation: a multidisciplinary approach, John Wiley & Sons, 2011.

[65] J. Banks. Principles of simulation. Handbook of simulation 12 (1998) 3–30.

[66] Z. Zhang, D. Wang, H. Yang, S. Si. A review of sequential decision making via simulation. arXiv preprint arXiv:2312.04090 (2023).

[67] J. P. Kleijnen. Verification and validation of simulation models. European journal of operational research 82 (1995) 145–162.

[68] R. G. Sargent, in: Proceedings of the 2010 winter simulation conference, IEEE, pp. 166–183.

[69] J. S. Carson, in: Proceedings of the winter simulation conference, volume 1, IEEE, pp. 52–58.

[70] Y. Laili, L. Zhang, Y. Luo. A pattern-based validation method for the credibility evaluation of simulation models. Simulation 96 (2020) 151–167.

[71] Z. Liu, L. Lai, L. Zhang. An empirical learning-based validation procedure for simulation workflow. arXiv preprint arXiv:1809.04441 (2018).

[72] W. Blog, Rag evaluation, 2024. URL: https://weaviate.io/blog/rag-evaluation, accessed: 2025-05-16.

[73] J. Luftman, in: Strategies for information technology governance, IGI Global Scientific Publishing, 2004, pp. 99–128.

[74] Y. E. Chan, B. H. Reich. It alignment: what have we learned? Journal of Information technology 22 (2007) 297–315.

[75] C. M. Macal, M. J. North, in: Proceedings of the 2009 winter simulation conference (WSC), IEEE, pp. 86–98.

[76] K. Ougaabal, G. Zacharewicz, Y. Ducq, S. Tazi. Visual workflow process modeling and simulation approach based on non-functional properties of resources. Applied Sciences 10 (2020) 4664.

[77] G. S. Fishman, Discrete-event simulation: modeling, programming, and analysis, volume 537, Springer, 2001.

[78] A. Ponomarenko, N. Avrelin, B. Naidan, L. Boytsov. Comparative analysis of data structures for approximate nearest neighbor search. Data analytics (2014) 125–130.

[79] Y. A. Malkov, D. A. Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. IEEE transactions on pattern analysis and machine intelligence 42 (2018) 824–836.

[80] G. Salton. Automatic text processing: The transformation, analysis, and retrieval of. Reading: Addison-Wesley 169 (1989).

[81] V. Ashish. Attention is all you need. Advances in neural information processing systems 30 (2017) I.

[82] R. Sennrich. Neural machine translation. Institute for Language, Cognition and Computation University of Edinburgh 18 (2016).

[83] N. Reimers, I. Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. arXiv preprint arXiv:1908.10084 (2019).

[84] T. Mikolov, K. Chen, G. Corrado, J. Dean. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013).

[85] A. Singhal, et al. Modern information retrieval: A brief overview. IEEE Data Eng. Bull. 24 (2001) 35–43.

[86] D. Cer, Y. Yang, S.-y. Kong, N. Hua, N. Limtiaco, R. S. John, N. Constant, M. Guajardo-Cespedes, S. Yuan, C. Tar, et al. Universal sentence encoder. arXiv preprint arXiv:1803.11175 (2018).

[87] G. Salton, A. Wong, C.-S. Yang. A vector space model for automatic indexing. Communications of the ACM 18 (1975) 613–620.

[88] J. Mu, S. Bhat, P. Viswanath. Representing sentences as low-rank subspaces. arXiv preprint arXiv:1704.05358 (2017).

[89] S. Robertson, H. Zaragoza, et al. The probabilistic relevance framework: Bm25 and beyond. Foundations and Trends® in Information Retrieval 3 (2009) 333–389.

[90] S. E. Robertson, K. S. Jones. Relevance weighting of search terms. Journal of the American Society for Information science 27 (1976) 129–146.

[91] S. P. Harter, A PROBABILISTIC APPROACH TO AUTOMATIC KEYWORD INDEXING., Ph.D. thesis, The University of Chicago, 1974.

[92] G. Zuccon, B. Koopman, P. Bruza, L. Azzopardi, in: Proceedings of the 20th Australasian document computing symposium, pp. 1–8.

[93] Y. Bachrach, Y. Finkelstein, R. Gilad-Bachrach, L. Katzir, N. Koenigstein, N. Nice, U. Paquet, in: Proceedings of the 8th ACM Conference on Recommender systems, pp. 257–264.

[94] J. Shawe-Taylor, N. Cristianini, Kernel methods for pattern analysis, Cambridge university press, 2004.

[95] B. Schölkopf, A. J. Smola, Learning with kernels: support vector machines, regularization, optimization, and beyond, MIT press, 2002.

[96] I. Goodfellow, Deep learning, 2016.

[97] J. Johnson, M. Douze, H. Jégou. Billion-scale similarity search with gpus. IEEE Transactions on Big Data 7 (2019) 535–547.

[98] H. Jegou, M. Douze, C. Schmid. Product quantization for nearest neighbor search. IEEE transactions on pattern analysis and machine intelligence 33 (2010) 117–128.

[99] A. Babenko, V. Lempitsky. The inverted multi-index. IEEE transactions on pattern analysis and machine intelligence 37 (2014) 1247–1260.

[100] A. K. Jain, M. N. Murty, P. J. Flynn. Data clustering: a review. ACM computing surveys (CSUR) 31 (1999) 264–323.

[101] V. Karpukhin, B. Oguz, S. Min, P. S. Lewis, L. Wu, S. Edunov, D. Chen, W.-t. Yih, in: EMNLP (1), pp. 6769–6781.

[102] L. Xiong, C. Xiong, Y. Li, K.-F. Tang, J. Liu, P. Bennett, J. Ahmed, A. Overwijk. Approximate nearest neighbor negative contrastive learning for dense text retrieval. arXiv preprint arXiv:2007.00808 (2020).

[103] T. Chen, S. Kornblith, M. Norouzi, G. Hinton, in: International conference on machine learning, PmLR, pp. 1597–1607.

[104] The Open Group, ArchiMate® 3.0.1 Specification, The Open Group, 2017. URL: https://pubs.opengroup.org/architecture/archimate3-doc/.

[105] R. Lu, S. Sadiq, in: International conference on business information systems, Springer, pp. 82–94.

[106] W. M. Van der Aalst. Business process management: a comprehensive survey. International Scholarly Research Notices 2013 (2013) 507984.

[107] F. Bachmann, L. Bass, J. Carriere, P. C. Clements, D. Garlan, J. Ivers, R. Nord, R. Little. Software architecture documentation in practice: Documenting architectural layers (2000).

[108] G. Poels, F. García, F. Ruiz, M. Piattini. Architecting business process maps. Computer Science and Information Systems 17 (2020) 117–139.

[109] R. Van de Wetering. Dynamic enterprise architecture capabilities and organizational benefits: an empirical mediation study. arXiv preprint arXiv:2105.10036 (2021).

A

Appendix A: Systematic Literature Review

Table A1: Table reporting XAI articles that have been examined to conduct the research and highlighting their main features

Author	Settings	Main Purpose	Technique[1]	Methods used for evaluation
Hennekeuser et al. (2024) [10]	Higher education lecturing environments	To develop and assess an LLM-based assistant with RAG capabilities for supporting lecturers	Implementation of an LLM-based assistant utilizing RAG with lecturing materials as the data foundation	Lecturer readiness and system reliability, explainability, and trustworthiness
Vera et al. (2024) [7]	Educational environments focusing on Rubik's Cube solving	To introduce ALLURE, an XAI-driven multimodal AI platform to enhance learner self-efficacy	Development of an XAI chatbot within the ALLURE platform	–
Holmes et al. (2022) [5]	AIED research community	To address ethical issues by proposing a multidisciplinary framework	Survey of 60 AIED researchers, with 17 contributing insights	–
Kasneci et al. (2023) [27]	Educational applications of LLMs	To present potential benefits and challenges of LLMs in education	Position paper on current applications and implications	–
Renz et al. (2021) [28]	Educational technology development	Introduce human-centered AI aligned with human values	Conceptual analysis of human-centered AI	Export evaluation, article reviews, user feedback
Bhat et al. (2024) [29]	Adults without technical backgrounds	To develop three tools for AI literacy and transparency	Design and implementation of web-based tools	Number of interactions, assessments, NPS
Hooshyar et al. (2024) [15]	Educational AI systems	Evaluate SHAP, LIME vs. neural-symbolic rule extraction	Comparative study of explanation techniques	Alignment with ground truth, comprehension, accuracy change
Chaushi et al. (2023) [30]	Educational technology development	Review of current XAI research and educational applications	Literature review	Identification of domains, challenges, opportunities
Jafarzade et al. (2023) [31]	K-12, higher ed, online platforms	Explore GPT in education: bias, misuse, accessibility	Literature review and analysis	–
Uglev et al. (2023) [32]	ITS in STEM subjects	Develop a dialogue system to enhance understanding	Implementation of explanatory dialogue strategies	–
Fiok et al. (2022) [11]	K-12, higher ed, and online platforms	Review capabilities and limitations of XAI in education	Literature review	–
Swamy et al. (2022) [13]	MOOCs	Compare LIME, SHAP, DiCE, CEM for success prediction	Applied to BiLSTM models	PCA, Jensen-Shannon distance, Spearman correlation
Khosravi et al. (2022) [2]	K-12, higher ed	Review XAI tools for education and training	Literature review	–
Yang et al. (2021) [16]	K-12, higher ed, online learning	Explore human-centered AI's role in education	Literature review and analysis	–
Saarela et al. (2024) [8]	Healthcare, finance, education	Systematic review of recent XAI applications	PRISMA-based literature review	Categorization of domains, XAI techniques
Kostopoulos et al. (2024) [23]	Healthcare, finance, education	Systematic review of XAI in DSSs	PRISMA-based literature review	–
Casalino et al. (2023) [26]	K-12, higher ed, online platforms	Analyze XAI tools to assist educational users	Literature review	–
Rorseth et al. (2024) [9]	Research and AI development	Enhance LLM transparency by tracing provenance	RAGE tool to trace knowledge source influence	User studies for usability and accuracy
Kaur et al. (2022) [4]	Education, Healthcare	Review principles for trustworthy AI	Systematic literature review	Fairness, transparency, bias reduction, etc.

[1] Note: LLM refers to "Large Language Model, RAG refers to "Retrieval-Augmented Generation", XAI refers to "Explainable Artificial Intelligence", LIME stands for "Local Interpretable Model-agnostic Explanations", SHAP refers to "SHapley Additive exPlanations,", PT refers to "Generative Pre-Trained Transformer,", ITS refers to "Intelligent Tutoring Systems,", DiCE refers to "Diverse Counterfactual Explanations,", CEM stands for "Counterfactual Explanations with Minimal Changes,"

Duan et al. (2024) [6]	Undergraduate CS course	Create trustworthy models with instructor involvement	XAI with human-AI collaboration	MAE, RMSE, pre/post comparison
Wang et al. (2024) [12]	STEM classrooms	Evaluate impact of model explanations on trust and load	Experimental design with surveys	Trust Scale, TAM, Cognitive Load
Ogata et al. (2024) [17]	K-12, higher ed, online platforms	Develop dual-explanation learning systems	Self- and AI-generated explanations	Engagement, learning improvement, satisfaction
Kamawar et al. (2024) [18]	AI development settings	Introduce Interactive Process Modeller (IPM)	Tool for modeling inter-agent relationships	Expert reviews, logs
Liu et al. (2024) [25]	K-12, higher ed, online learning	Overview of empirical XAI in education	Systematic literature review	–
Tiddi et al. (2022) [19]	Finance, education	Use of knowledge graphs in XAI	Systematic literature review	–
Chou et al. (2022) [20]	Healthcare, finance, education	Systematic review of counterfactual XAI algorithms	PRISMA with LDA topic modeling	–
Cambria et al. (2023) [21]	Finance, education	Review 70 XAI papers (2006–2021)	Hierarchical MCDM-based literature review	–
Zerilli et al. (2022) [3]	Finance, education	Examine transparency's impact on trust	Literature review and analysis	–
Gajos et al. (2022) [14]	Lab experiments in nutrition decisions	Study effect of AI assistance on engagement	Experimental design with 3 conditions	Decision accuracy, engagement, learning
Papenmeier et al. (2022) [22]	Online study with 959 participants	Study trust based on accuracy and explanation type	User trust evaluation across classifiers	Likert, logs, surveys

Table A2: Table reporting EA articles that have been examined to conduct the research and highlighting their main features

Author	Settings	Main Purpose	Technique[2]	Methods used for evaluation
Rittlemeyer et al. (2021)[34]	General enterprise architecture literature	To investigate how AI adoption influences different layers of enterprise architecture through a systematic literature review	Structured review focusing on architectural impacts such as business processes, application layers, and technology infrastructure	Literature-based synthesis, framework development for identifying change areas
Kamalabai et al. (2024)[36]	Theoretical enterprise architecture and AI governance frameworks	To advocate for sustainable EA approaches that align with ethical and long-term AI integration	Conceptual modeling of EA with sustainability indicators and policy alignment mechanisms for AI integration	Conceptual framework evaluation using design principles and qualitative justification
Sayles et al. (2024)[38]	Enterprise AI governance contexts across industries	To explore how EA can guide ethical AI deployment by offering transparency and control in complex systems	Discussion of EA structures supporting XAI, model governance, and organizational alignment strategies	Case-based discussion and theoretical analysis linking EA patterns to responsible AI governance
Ilin et al. (2019)[33]	Digital transformation initiatives in enterprises	To present enterprise architecture modeling approaches supporting organizational change during digital transformation	Development of a meta-model demonstrating the impact of digital technologies on various enterprise architecture components	Case studies illustrating the application of the meta-model in real-world digital transformation scenarios
Dilnutt et al. (2023)[39]	Manufacturing industry undergoing digital transformation	To examine how Enterprise Architecture (EA) supports the integration of Artificial Intelligence (AI) in digital transformation initiatives within the manufacturing sector	Analysis of EA frameworks to identify their role in facilitating AI adoption and addressing complexities in smart manufacturing	Case studies from the manufacturing industry illustrating EA's impact on AI-driven digital transformation outcomes
Zimmermann et al. (2020)[40]	Enterprises undergoing digital transformation	To explore the evolution of enterprise architecture in supporting intelligent digital systems amidst advancements in artificial intelligence and digitalization	Analysis of digital platform architectures and ecosystems, emphasizing the integration of AI, IoT, and cloud computing within enterprise architecture frameworks	Theoretical discussion supported by literature review, presenting methodological advances and evolutionary paths for digital enterprise architectures
Malott et al. (2024)[35]	Organizations undergoing AI-driven digital transformation	To discuss the role of Enterprise Architecture (EA) in facilitating AI-powered digital transformation within organizations	Exploration of EA frameworks and strategies to effectively integrate AI technologies into business processes, enhancing efficiency and innovation	Theoretical analysis supported by industry case studies demonstrating successful AI integration through robust EA practices
Fitriani et al. (2023)[37]	Organizations adopting AI in business processes	To develop a framework integrating TOGAF and BIZBOK for AI adoption in business architecture, aiming to enhance process efficiency, innovation, and resource optimization	Combination of TOGAF's ADM methodology and BIZBOK guidelines to structure AI integration across business capabilities, value streams, and organizational structures	Framework validation through case studies demonstrating improved AI adoption in business architecture, enhancing operational efficiency and strategic alignment
Denni et al. (2024)[41]	Organizations integrating Generative AI into enterprise architecture	To identify critical success factors for adopting and strategically integrating Generative AI (GenAI) within enterprise architecture, aiming to enhance efficiency, productivity, and decision-making	Examination of factors such as responsible AI development, data governance, stakeholder collaboration, and adherence to international regulations to build trust and minimize risks in GenAI adoption	Theoretical analysis supported by discussions on evolving roles of enterprise architects and IT professionals in the AI-driven landscape
Wang et al. (2020)[42]	Large enterprises implementing enterprise analytics systems	To provide a comprehensive guide on integrating advanced analytics into data-driven business strategies within large enterprises	Methodological framework combining strategic, architectural, and managerial aspects of enterprise analytics, including data management, cloud platforms, and AI solutions	Case studies and practical applications demonstrating the successful implementation of enterprise analytics to foster growth and optimize operations

[2]Note: EA refers to "Enterprise Architecture"

Sandkuhl et al. (2021)[43]	Organizations implementing AI applications	To investigate how Enterprise Architecture (EA) models can support requirements engineering and assess organizational readiness for AI adoption	Analysis of various AI application types and their specific requirements within organizational IT landscapes, utilizing EA models for context analysis	Literature review, case study analysis, and method development for AI context analysis based on EA models
Martel et al. (2021)[44]	Organizations integrating AI into enterprise systems	To identify obstacles and propose best practices for integrating data science and AI within enterprise architectures, aiming to enhance collaboration, accelerate deployment, and standardize operations	Adoption of an integrated approach combining best practices from data science and DevOps, facilitating efficient governance and collaboration across development cycles	Theoretical analysis supported by case studies demonstrating improved AI integration and operational efficiency within enterprise architectures
Takeuchi et al. (2019)[45]	AI service systems development in enterprises	To propose an enterprise architecture (EA)-based modeling approach for developing AI service systems, emphasizing the importance of integrating AI within organizational structures	Use of EA modeling techniques to design AI service systems that align with business processes, ensuring scalability, and system integration	Evaluation through conceptual modeling, focusing on the effectiveness of AI system integration within organizational frameworks [45]

Table A3: Table reporting publications comparing EA frameworks

Author	Settings	Main Purpose	Technique	Methods Used for Evaluation
Martel et al. (2021)[44]	Enterprise AI integration	Identify best practices for integrating AI into enterprise software architecture	Analysis of architectural considerations for scalable and maintainable AI deployment	Case studies and theoretical analysis
Takeuchi & Yamamoto (2019)[45]	AI service system development	Propose an EA-based modeling approach for AI service systems	Utilization of EA modeling techniques to align AI services with organizational processes	Conceptual modeling and application to project analysis
Abdallah et al. (2021)[47]	EA measurement literature	Systematic review of EA measurement approaches	Identification of key metrics and evaluation methods for EA effectiveness	Literature-based synthesis and framework development
Urbaczewski & Mrdalj (2006)[50]	Comparison of EA frameworks	Analyze structures and methodologies of various EA frameworks	Comparative analysis of frameworks like TOGAF and Zachman	Theoretical comparison and analysis
Singh & Mudholkar (2015)[51]	EA frameworks study	Comparative study of TOGAF and Zachman's EA frameworks	Evaluation of strengths and weaknesses of each framework	Theoretical analysis and recommendations
Kotusev (2021)[59]	Top EA frameworks	Critical examination of four prominent EA frameworks	Analysis of practical value and implementation challenges	Theoretical critique and industry observations
Rouhani et al. (2013)[60]	EA implementation methodologies	Compare various EA implementation methodologies	Evaluation based on processes, tools, and outcomes	Comparative analysis and methodology assessment
Lolo et al. (2020)[46]	Educational services	Apply TOGAF framework to educational institutions	Development of comprehensive architecture encompassing business, data, application, and technology layers	Case studies from Universities and blueprint development

Author	Settings	Main Purpose	Technique	Methods Used for Evaluation
O'Higgins (2023)[49]	Digital transformation	Investigate the role of business architecture in digital transformation	Use of Balanced Scorecard and Structural Equation Modeling (SEM)	Empirical study with survey data and PLS-SEM analysis
Missikoff (2022)[48]	Business process analysis	Introduce a Business Process Analysis Canvas for early expert involvement	Structured approach to capturing domain knowledge	Methodology proposal and theoretical grounding
Poels et al. (2020)[108]	Business process mapping	Present methodologies for creating effective business process maps	Identification of inefficiencies and areas for improvement	Theoretical framework and process optimization strategies
Dresch et al. (2015)[61]	Design Science Research	Provide comprehensive overview of DSR methodology	Guidance for developing innovative solutions through iterative design	Literature review and methodological framework
Van de Wetering (2021)[109]	Dynamic EA capabilities	Explore how dynamic EA capabilities contribute to organizational benefits	Development of research model explaining EA capabilities' impact	Empirical study with survey data and mediation analysis

Query Nr	Query
1	("Explainable AI" OR "XAI" OR "Interpretability") AND ("Large Language Models" OR "RAG" OR "Transformer Models") AND ("Educational Environments" OR "Learning Platforms")
2	("Explainability" OR "Interpretability") AND ("Proactive Chatbots" OR "Conversational Agents") AND ("RAG" OR "Knowledge-Augmented Generation")
3	("Explainable AI" OR "XAI") AND ("RAG" OR "Knowledge-Augmented Models") AND ("Educational Tools" OR "Learning Management Systems")
4	("Data Science" OR "Information Retrieval" OR "Machine Learning Models") AND ("RAG" OR "Knowledge-Augmented Models") AND ("Chatbots" OR "Conversational Agents")

Table A4: XAI - Used Search Queries

Query Nr	Query
1	("Artificial Intelligence" OR "AI" OR "AI Integration") AND ("Enterprise Architecture" OR "EA" OR "Business Architecture") AND ("Frameworks" OR "Models" OR "Methodologies")
2	("Generative AI" OR "GenAI") AND ("Enterprise Architecture") AND ("Strategic Integration" OR "AI Adoption") AND ("Success Factors" OR "Evaluation")
3	("Enterprise Analytics" OR "Data Management" OR "AI Solutions") AND ("Large Enterprises") AND ("AI Integration" OR "Business Transformation")
4	("AI-based Systems" OR "AI Applications") AND ("Enterprise Architecture" OR "IT Infrastructure") AND ("Modeling" OR "Meta-modeling")

Table A5: EA - Used search Queries

Table A.6: XAI Implementation Strategies in Education

Strategy	Implementation Steps
Developing an XAI Integration Framework	• Define clear objectives and align XAI tools with educational goals. • Establish cross-functional teams for XAI governance. • Develop a phased roadmap for XAI deployment.
Infrastructure and Resource Allocation	• Assess and upgrade existing technological infrastructure. • Allocate financial and human resources for XAI implementation. • Utilize cloud-based and on-premise XAI solutions.
Training and Professional Development	• Implement XAI literacy programs for educators and staff. • Develop competency frameworks to integrate XAI into teaching. • Conduct continuous workshops, training, and certifications.
Data Governance and Ethical XAI Usage	• Establish ethical guidelines for responsible XAI deployment. • Implement data privacy and security policies. • Monitor XAI decision-making to prevent biases.

Continued on next page

Strategy	Implementation Steps
Pilot Testing and Iterative Implementation	• Conduct small-scale XAI pilot programs in different educational settings. • Collect stakeholder feedback for iterative improvements. • Scale up successful XAI implementations based on results.
Long-Term Monitoring and Improvement	• Establish XAI performance evaluation metrics. • Regularly update XAI models based on evolving educational needs. • Encourage research and innovation in XAI for education.

B

APPENDIX B: METHODOLOGY

B.1. DSR

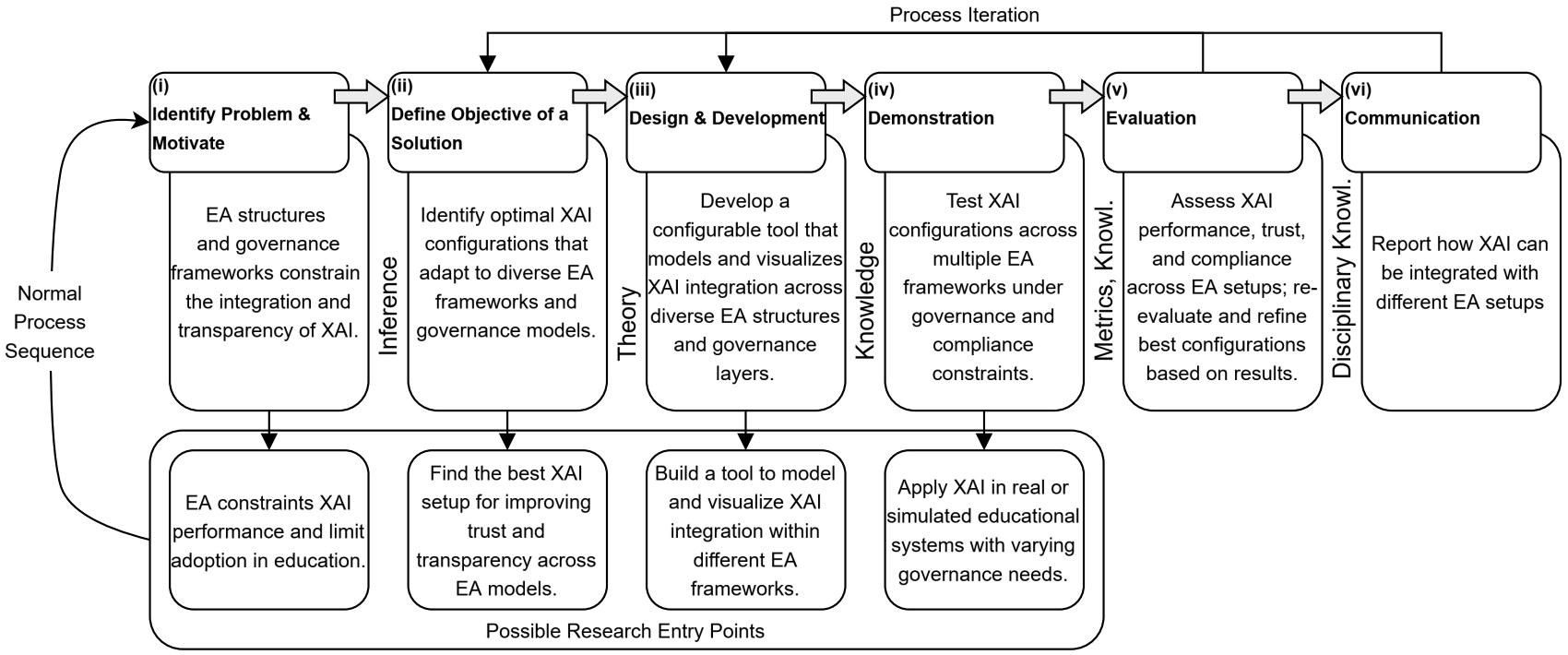


Figure B.1: DSR Cycle - Comparing different EA Architectures

B.2. CRISP-ML(Q)

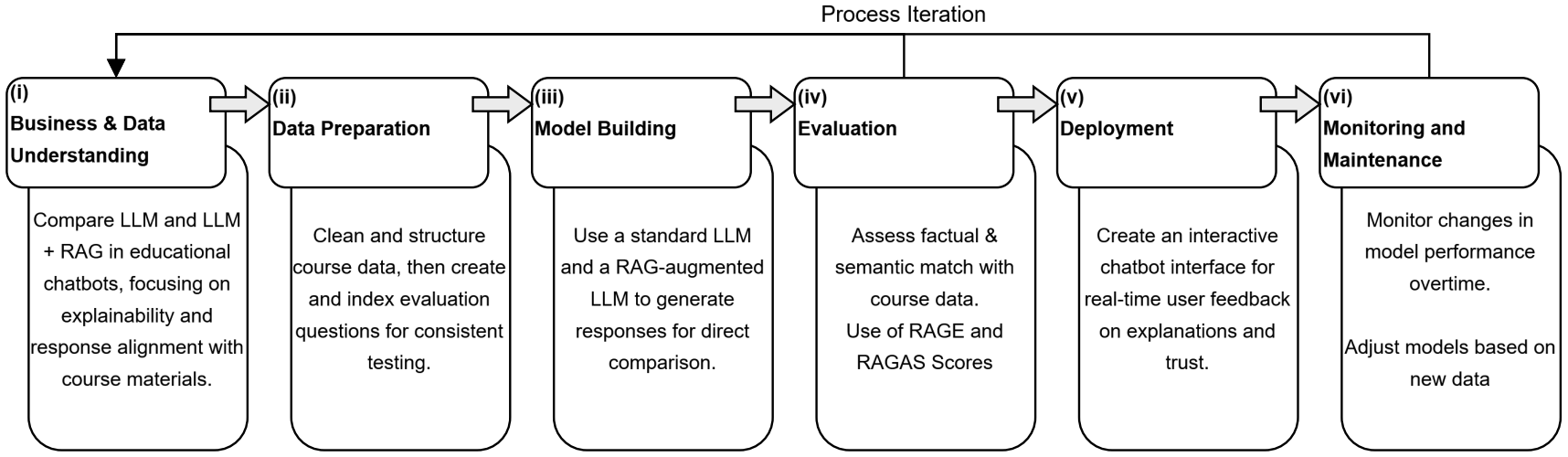


Figure B.2: CRISP ML - Comparing LLM and LLM+RAG in different setups

B.3. MODES-EA

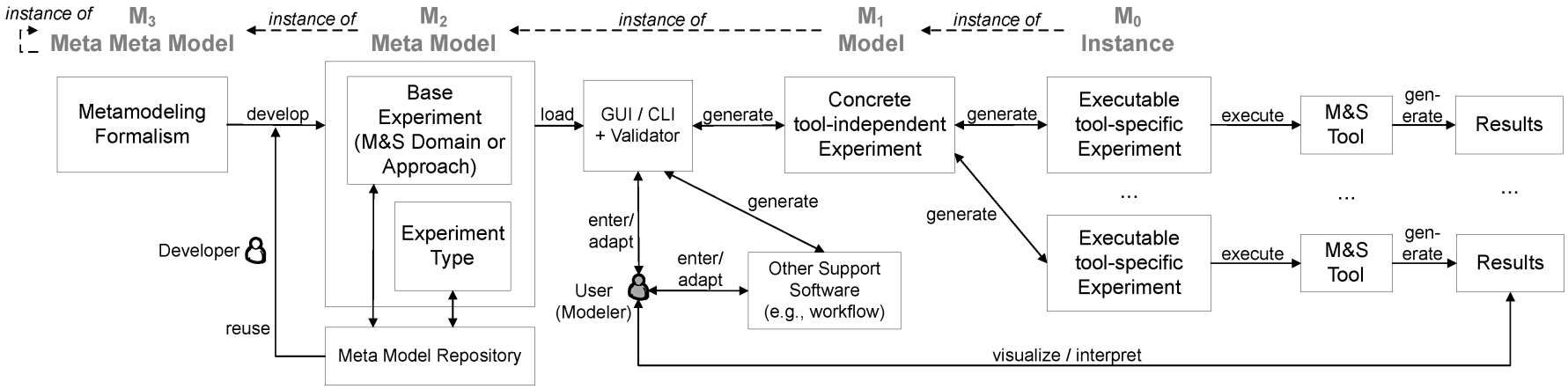


Figure B.3: Layered MDE Framework [1]

Table B.1: Summary - Layered MDE Framework by Wilsdorf[1]

Layer	Description
M3 – Meta-Meta Model	Defines the language used to specify meta-models. Typically includes constructs such as classes, relationships, and constraints for building modeling languages.
M2 – Meta-Model	Describes the domain-specific modeling language (DSML), including core modeling constructs such as entities, resources, events, and experimental setups.
M1 – Model	A user-defined model that conforms to the meta-model. Represents a specific DES system, including its structure, behaviors, and parameter settings.
M0 – Runtime Layer / Execution Artifacts	The operational realization of the M1 model. It includes generated simulation code and executable workflows that run on simulation platforms.

Table B.2: MODES-EA Modifications of M3 Layer

Modification	Reason	Description	Application
Support for multi-layered EA frameworks	Enable modeling of complex EA like TOGAF	Introduced meta-meta classes to represent hierarchical relationships between business, application, and technology layers	Allows structured representation of layered architectures within simulation models by introducing a translation layer, mapping concepts between the two disciplines
Formalized cross-layer constraints	Ensure consistency and compliance across architectural layers	Defined rules to enforce valid dependencies between elements from different EA layers	Enables automated validation of architectural integrity during simulation via rule enforcements and class constraints

Table B.3: MODES-EA Modifications of M2 Layer

Modification	Reason	Description	Application
EA-specific architectural elements	To represent key EA constructs such as application components, data stores, and infrastructure nodes that are essential for accurate architectural modeling.	Extended the meta-model with classes capturing architectural components and resources, aligning with common EA frameworks.	Supports detailed EA modeling by enabling simulation inputs to explicitly reference components, data flows, and infrastructure elements, facilitating realistic scenario analysis.
Process flows with control and data dependencies	To capture dynamic behavior and interactions within EA processes, reflecting real-world control sequences and data exchanges.	Integrated constructs to model both control flow and data flow within processes, supporting complex dependencies and coordination logic.	Allows simulation of process execution order and data-driven decisions, improving fidelity of resource and task modeling.
Organizational roles with resource attributes	To simulate the impact of human and organizational factors on EA behavior, including availability and prioritization of resources.	Added role entities with attributes such as skill level, availability schedules, and prioritization rules.	Enables fine-grained resource allocation and scheduling during simulation runs, reflecting organizational realities.
Queues and event triggers for EA events	To model asynchronous behavior and event-driven interactions commonly present in EA environments.	Incorporated queue constructs for resource contention and event trigger elements to capture event-based state changes.	Facilitates simulation of event-driven workflows, resource bottlenecks, and inter-component signaling within the architectural model.

Table B.4: MODES-EA Modifications of M1 Layer

Modification	Reason	Description	Application
Parameterized simulation models for EA scenarios	To reflect actual system usage patterns and variability in workload conditions.	M1 models are enriched with runtime parameters like user arrival rates, transaction volumes, and processing delays.	Enables experimentation with different architectural configurations, stress-testing systems under peak loads, and forecasting performance under varying conditions.
Workflow modeling with conditional logic	To simulate real-world decision-making processes and dynamic execution paths.	Implemented branching mechanisms at workflow nodes to represent business rules and conditional flows.	Allows assessment of alternate process paths and their implications on resource usage, throughput, and timing.
Resource instantiation from architectural roles	To model the behavior of human and technical actors within the simulated EA context.	Concrete instantiations of roles from the M2 layer are mapped to agents with specific schedules, capacities, and constraints.	Supports analysis of workload distribution, idle time, and the impact of staffing or infrastructure changes.
Simulation of infrastructure behavior	To evaluate how deployed architectural elements affect execution dynamics and system performance.	Application components are instantiated as service nodes with performance profiles, including latency, queue length, and reliability factors.	Enables detailed modeling of server pools, load balancing, and service degradation under strain or failure conditions.

Table B.5: MODES-EA Modifications of M0 Layer

Modification	Reason	Description	Application
Standardized simulation interface layer	To enable compatibility across different DES platforms and reduce integration complexity.	Introduces abstracted interface definitions for simulation input/output handling and execution control.	Supports switching between engines like AnyLogic or SimPy without rewriting model logic; enables platform-agnostic execution.
Metadata capture and logging infrastructure	To ensure transparency, traceability, and reproducibility of simulation runs.	Structured metadata is captured for each execution, including input parameters, timestamped logs, error reports, and run IDs.	Enables audit trails, automated experiment comparisons, and compliance with scientific reproducibility standards.
Execution artifact generation pipelines	To automate deployment of M1 models into runnable formats suited for simulation engines.	Model transformations and exporters generate scripts, configurations, and runtime code compatible with target engines.	Reduces manual workload; ensures consistent mapping between EA specifications and operational simulation logic.
Support for batch and sensitivity analysis execution	To facilitate systematic exploration of parameter effects and design alternatives.	Scripts and tools are created to run multiple scenarios with varying inputs, capturing aggregate outputs.	Enables performance benchmarking, stress testing, and optimization studies across EA configurations.

B.4. SIMULATION VERIFICATION AND VALIDATION

Table B.6: Validation Techniques for Simulation Models

Technique	Description / Purpose
Hypothesis Testing	Statistical comparison of simulated outputs with empirical or benchmark data using tests such as paired t-tests or ANOVA.
Confidence Interval Estimation	Quantifies uncertainty by constructing intervals for average outputs like throughput, delay, or resource utilization.
Regression-Based Metamodeling	Uses regression analysis to model relationships between input factors and outputs, aiding in validation and sensitivity analysis [67].

B.5. SIMULATION EVALUATION

Table B.7: Verification Procedures for Simulation Models

Procedure	Description / Purpose
Test Case Debugging	Involves running the model on simplified or edge-case inputs to verify logic correctness and detect computational errors.
Structured Model Walkthroughs	Step-by-step evaluation of the model structure and assumptions, often conducted collaboratively with stakeholders.
Code Inspection	Line-by-line examination of model logic and code to ensure intended behavior and detect logic flaws [69].

Table B.8: Sandkuhl and Rittelmeyer - Evaluation Process for EA Models

Evaluation Step	Purpose	Key Considerations
Literature and Context Review	Examine existing EA literature and contextual background for AI integration.	Identifies limitations in current frameworks, uncovers challenges to AI integration, and ensures EA reflects technological demands like data handling and infrastructure.
Classification of AI Applications	Identify and categorize various types of AI applications and their needs.	Focuses on differences in data, computation, and organizational needs for applications such as ML, NLP, and expert systems.
EA Model Component Analysis	Analyze EA components for their ability to support AI implementation.	Evaluates infrastructure, scalability, flexibility, and support for workflows that are essential for AI adoption.
Alignment with AI Requirements	Compare EA capabilities against the requirements of AI applications.	Involves requirements engineering and assessing how well the EA integrates AI into existing business processes.
Organizational Readiness Assessment	Determine whether the organization is prepared for AI integration.	Assesses cultural, technical, and structural readiness, ensuring infrastructure and change management practices are in place.

C

APPENDIX C: TRUSTWORTHY ARTIFICIAL INTELLIGENCE

C.1. DATA DESCRIPTION

Table C.1: Structural Complexity Scale

Level	Description
Low	Flat or minimally structured content; simple formatting; limited use of sections or hierarchy.
Medium	Contains sections, headings, or limited media; some hierarchy present but manageable.
High	Deeply nested structures, rich formatting, figures, embedded media, and formal document divisions.

Table C.2: Information Density Scale

Level	Description
Low	General or superficial content; low conceptual granularity; often lacks theoretical precision.
Medium	Balanced level of detail; includes elaboration but with moderate redundancy.
High	Concise and concept-rich; high theoretical load per unit; minimal repetition.

Table C.3: Parsing Requirements Scale

Level	Description
Low	Direct extraction using standard parsers; plain text with minimal layout dependencies.
Medium	Requires structural parsing or tag-based extraction; includes formatting or basic media.
High	Complex synchronization of multimodal elements; non-trivial pre-processing and alignment.

Table C.4: Metadata Availability Scale

Level	Description
Low	No embedded metadata; only filename-level or implicit information.
Medium	Partial metadata available (timestamps, speaker IDs, authorship).
High	Rich metadata (DOIs, ISBNs, section labels, publication metadata, etc.).

Table C.5: Annotation Potential Scale

Level	Description
Low	Lacks clear semantic or structural units; difficult to segment or label.
Medium	Mixed structure; some annotatable units but with alignment challenges.
High	Clearly defined semantic units (chapters, slides, sections); annotation-friendly.

C.2. DATA RELATIONSHIP

Table C.6: Course Object Structure Description

Field	Description
id	Unique identifier for the course within the platform.
name	Human-readable title of the course.
accountID	Identifier for the account or organization owning the course.
uuid	Universally unique identifier for the course resource.
start at	Scheduled start date and time for the course.
is public	Visibility flag indicating if course is publicly accessible.
created at	Timestamp for course creation.
course code	Code or shorthand identifier for the course, often used for registration.
enrollment term id	Identifier representing the academic term or semester of the course.
license	Licensing or copyright status for the course materials.
end at	Scheduled end date and time for the course.
public syllabus	Flag indicating if syllabus is publicly visible.
storage quota mb	Storage limit allocated to the course in megabytes.
locale	Language and regional settings used in the course interface.
time zone	Time zone setting relevant to course scheduling and deadlines.
workflow state	Current operational status of the course (e.g., available, completed).

Table C.7: Module Structure Attributes

Attribute	Description
id	Unique identifier assigned to the module for internal referencing and database operations.
name	The title or label given to the module, describing its thematic content or role within the course.
position	Numerical order indicating the module's sequence within the overall course structure.
unlock at	Timestamp or condition specifying when the module becomes accessible to learners; may be null if unrestricted.
require sequential progress	Boolean flag indicating if learners must complete previous modules before accessing this one.
requirement type	Defines the criteria needed to fulfill the module's requirements (e.g., all items must be completed).
published	Boolean status indicating whether the module is visible and accessible to learners.
items count	Total number of learning items (e.g., lectures, quizzes) contained within the module.
items url	API endpoint or resource locator providing access to the detailed list of items within the module.

Table C.8: Structure of a Module Item (Field Descriptions)

Field	Description
id	Unique identifier for the module item.
title	Descriptive name or filename of the content associated with the item.
position	Numerical order of the item within the module sequence.
type	Classification of the item content type (e.g., File, Page, Quiz).
module id	Identifier of the parent module containing this item.
content id	Identifier linking to the underlying content resource (e.g., a file or page).

Field	Description
url	Direct API endpoint URL to retrieve or interact with the content.
published	Boolean indicating whether the item is publicly visible to users.

Table C.9: Module Data File Metadata Structure

Field	Description
id	Unique identifier for the file within the system.
folder id	Identifier of the folder containing the file, representing organizational hierarchy.
display name	Human-readable name of the file, used for UI display.
filename	Actual file name stored in the system, including extension.
upload status	Status indicating success or failure of file upload.
content type	MIME type specifying the file format (e.g., application/pdf).
url	Direct download URL for the file resource.
size	File size in bytes, indicating storage and transfer requirements.
created at	Timestamp when the file was originally created in the system.
updated at	Timestamp for the last update made to the file metadata or content.
unlock at	Timestamp when the file becomes accessible, if restricted.
locked	Boolean flag indicating if the file is locked for editing or viewing.
hidden	Boolean flag indicating if the file is hidden from standard views.
modified at	Timestamp indicating last modification date of the file content.
mime class	General class of the MIME type (e.g., pdf, image).
category	Classification label for the file content (e.g., uncategorized).

C.3. PROMPT CONSTRUCTION

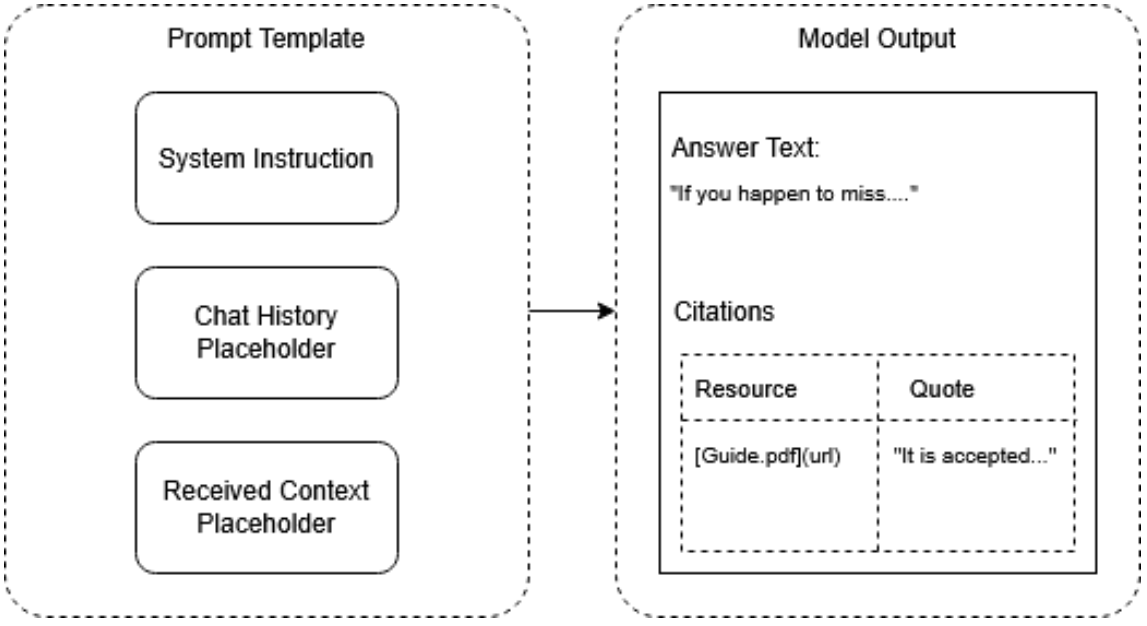


Figure C.1: TAI - Prompt Template and Output Structure

C.4. RAGE

Table C.10: Evaluation of Faithfulness and Source Attribution in RAG Systems

Evaluation Metric	Definition	Importance in Educational Applications
Input Faithfulness	Assesses whether the retrieved documents used as context for the language model are relevant and necessary to the query.	Ensures that the context provided to the model is appropriate, reducing the introduction of irrelevant or noisy information that may undermine user trust and the accuracy of responses. This is especially important in education, where the integrity of provided content is critical.
Output Faithfulness	Evaluates whether the generated answer stays true to the retrieved input, differentiating between factual correctness and factual grounding.	Ensures that the generated response aligns with the provided documents and does not introduce new, unsupported information. This is crucial for educational chatbots, as students must be able to trace responses back to credible and authoritative sources.
Source Attribution	Examines how and whether the model references or cites the retrieved context, supporting explainability and transparency.	Enhances user trust and reduces cognitive load by making it clear why a particular answer is valid. It is particularly important in educational contexts, where users (learners) need to understand the origin of the information provided to verify its accuracy and relevance.

C.5. LLM PROMPTS

C.5.1. QUESTION ANSWERING

```
You are an assistant for question-answering tasks.

Use the following pieces of retrieved context to answer the question.
You must use the context to formulate your answer, but do not mention or refer
 ↪  to the context itself.
Focus solely on answering the question accurately and concisely.
If you don't know the answer based on the provided information, respond with "I
 ↪  don't know."
Provide as detailed answers as possible.

Question: question
Context: context
Answer:
```

C.5.2. LLM METRIC PROMPT

```
You are evaluating an AI-generated answer to a question using the following
 ↪  information:

- The original question
- The retrieved context used to generate the answer
- The AI-generated answer
- The expected ground truth answer

Your task is to assess whether the answer is appropriate, informative, and
 ↪  aligned with the question and ground truth.
Do NOT expect the answer to copy the ground truth verbatim. Paraphrasing,
 ↪  restructuring, or simplification is acceptable - and even preferred - as
 ↪  long as the core meaning is preserved.
```

```
Answers are contextual and can vary in style and detail. You SHOULD NOT penalize
 ↪  answers for diverging in language from the ground truth.
Longer context is not a problem - more context often helps improve the answer,
 ↪  and longer responses should not be penalized.
If the model notes that the context was long or says it couldn't process the
 ↪  full input, ignore that - it should not lower the score.

When scoring, use these criteria:

- context_recall: Does the answer reflect the key ideas present in the context
 ↪  and ground truth, even if paraphrased?
- context_precision: Is the answer focused only on relevant content from the
 ↪  context and ground truth, avoiding speculation?
- faithfulness: Is the answer factually consistent with the content of the
 ↪  context and the ground truth?
- answer_relevancy: Does the answer meaningfully and directly address the
 ↪  question?

You are judging informativeness, focus, and factual accuracy - not literal
 ↪  overlap.
If the answer resembles or infers the ground truth's meaning in any reasonable
 ↪  way, scores below 0.5 are not appropriate.
Use the full range from 0.0 to 1.0, but reserve low scores only for factually
 ↪  wrong, irrelevant, or misleading answers.
Mostly correct answers should score between 0.7 and 1.0.

Respond ONLY in this JSON format:

    "context_recall": X.X,
    "context_precision": X.X,
    "faithfulness": X.X,
    "answer_relevancy": X.X


Question: question
Answer: answer
Ground Truth: ground_truth
```

C.5.3. RAGE PROMPT

```
You are evaluating a language model's answer to a question.

Given:

- The question
- The retrieved context used to generate the answer
- The final generated answer
- The ground truth answer
```

```
Evaluate the following:

1. Input Faithfulness - To what extent is the answer based on the context?
2. Output Faithfulness - Is the answer factually aligned with the ground truth?
3. Source Attribution - Can the answer be causally linked to specific content in
 ↪  the context?

Use the full 0.0-1.0 scale.

You are evaluating overall factual alignment and informativeness - not wording.
Paraphrasing, partial matches, and inferred correctness are valid.
If the answer is generally informative and resembles or touches on the key
 ↪  elements of the ground truth, give it a reasonably high score.
Small factual gaps, rewording, or imperfect structure should not result in low
 ↪  scores.

Respond only in JSON:

    "input_faithfulness": X.XX,
    "output_faithfulness": X.XX,
    "source_attribution": X.XX


Question: question
Context: context
Answer: answer
Ground Truth: ground_truth
```

D

APPENDIX D: ENTERPRISE ARCHITECTURE

D.1. ARTEFACT ARCHITECTURE

Table D.1: Flowchitect - Application Structure

Component Group	Subcomponent	Purpose	Contribution to Flowchitect
Application Core	components	Provides encapsulated UI and interaction units	Offers atomic building blocks for views and model representation
	composables	Encodes reusable logic for state and behavior	Structures model execution and simulation flows through shared logic patterns
	layouts	Organizes structural templates for views	Differentiates major application modes like simulation and editing
	middleware	Controls access routing based on state	Enforces rules before transitions across simulation stages
	pages	Defines user-facing navigation endpoints	Serves as the gateway to modelling and simulation environments
	plugins	Bridges external services and tools	Connects analytics, APIs, and shared configurations into app flow
	utils	Hosts shared helper routines	Facilitates internal processing for data shaping and control logic
Extension System	layers	Enables structured feature isolation	Supports injection of modular behaviors into core simulation logic
	modules	Integrates reusable application extensions	Adds encapsulated features like state saving and visual tooling
Backend Layer	api	Delivers endpoints for client interaction	Processes simulation requests and manages data persistence
	middleware	Applies logic on server requests	Controls validation and security during simulations
	plugins	Provides backend utility injection	Adds tools for handling sessions and server configuration
	routes	Structures server-side path logic	Organizes backend feature access and logic routing
	utils	Centralizes backend helpers	Supports data formatting, logs, and simulation computation

D.2. ARTEFACT PRIMITIVES

Table D.2: Flowchitect - Node Structure

Node Characteristic	Behavior	Technical Definition of Enabling Attribute	System Impact and Functional Role
Fundamental Identity Management	Unique Identification	Achieved via a unique string identifier assigned to each node instance.	Serves as the primary key for node referencing, enabling programmatic access and graph traversal.
	Functional Classification	Defined by a string specifying the node's operational category, which can be a predefined or custom value.	Directs rendering logic and influences default handle placement for connections.
Spatial and Geometric Control	Positional Control	Governed by Cartesian coordinates representing the node's top-left corner within the canvas.	Defines node placement, used for layout algorithms and dynamic graph visualization.
	Explicit Sizing	Controlled by an optional object defining the node's fixed dimensions in pixels .	Provides precise control over the node's bounding box for layout calculations and collision detection.
Internal State Encapsulation	Data Encapsulation	Managed by a generic JavaScript object for storing domain-specific state or operational parameters.	Enables encapsulation of node-specific computational data, reactively propagating updates throughout the Vue system.
Structural and Hierarchical Organization	Hierarchical Grouping	Established by referencing the id of a parent node (attribute: parentNode?), creating a nested relationship.	Organizes nodes into modular units, supporting complex multi-level flow architectures.
	Movement Constraint	Defined by an optional argument that restricts a child node's movement within a specified boundary.	Enforces spatial integrity, programmatically limiting child node movement to preserve hierarchical structure.
	Z-Axis Layering	Controlled by an optional numerical value determining the node's stacking order.	Manages visual occlusion, ensuring critical nodes maintain visual precedence in overlapping scenarios.
Connection Point Configuration	Incoming Connection Behavior	Configured by an optional enumeration for default incoming edge attachment .	Provides a default point of entry for edges, influencing connection rendering and validation.
	Outgoing Connection Behavior	Configured by an optional enumeration for default outgoing edge attachment.	Provides a default point of origin for edges, influencing connection rendering and validation.

Table D.3: Flowchitect - Edge Structure

Edge Characteristic	Behavior	Technical Definition of Enabling Attribute	System Impact and Functional Role
Fundamental Identity and Connectivity	Unique Identification	Achieved via a unique string identifier assigned to each edge instance.	Serves as the primary key for edge indexing, enabling programmatic access, state management, and graph traversal algorithms.
	Source Node Association	Defined by a string representing the id of the originating node for the connection .	Establishes the directional start point of the edge, critical for graph topology and data flow definition.
	Target Node Association	Defined by a string representing the id of the destination node for the connection .	Establishes the directional end point of the edge, completing the connectivity definition within the graph.
	Specific Source Endpoint Definition	Configured by an optional string identifying a specific handle on the source node.	Enables precise connection to a designated anchor point on the source node, allowing multiple distinct connections from a single node.
	Specific Target Endpoint Definition	Configured by an optional string identifying a specific handle on the target node.	Enables precise connection to a designated anchor point on the target node, allowing multiple distinct connections to a single node.
Connection Type and Data	Connection Type Specification	Defined by a string categorizing the edge's rendering and behavior.	Dictates the visual curvature and rendering path of the edge, and can influence custom logic or validation rules.
	Data Encapsulation	Managed by a generic JavaScript object for storing custom state or metadata specific to the connection.	Enables the aggregation of arbitrary computational data or properties associated with the edge, reacting to updates within the Vue system.
Dynamic Visual and State Behaviors	Dynamic Flow Visualization	Controlled by a boolean that, when true, renders a subtle animation along the edge .	Provides visual cues for active data flow or ongoing processes, enhancing the interpretability of dynamic graphs.

Edge Characteristic	Behavior	Technical Definition of Enabling Attribute	System Impact and Functional Role
	Updatable Connection	Governed by a boolean that, when true, allows the user to re-route or modify the edge's source/target connections.	Enables interactive manipulation of the graph topology by users, facilitating dynamic re-configuration of connections.

D.3. SIMULATION DATA STRUCTURE

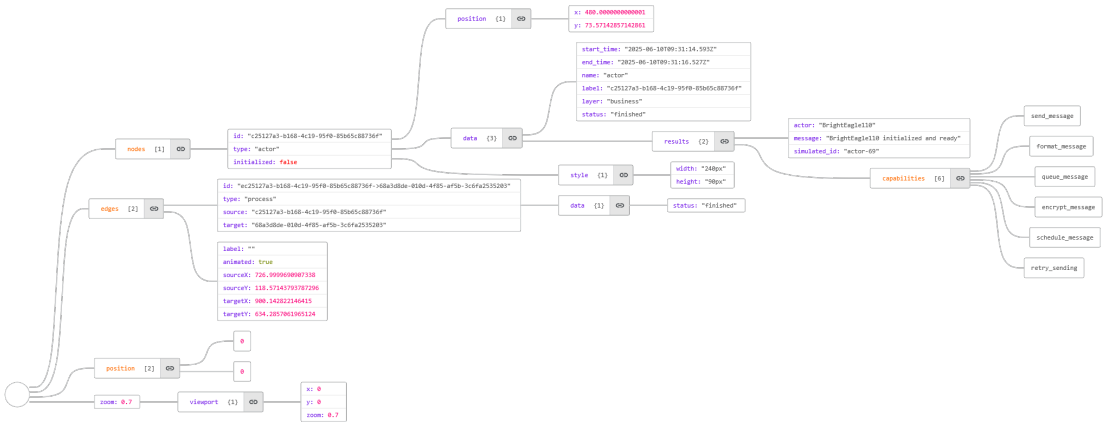


Figure D.1: Data Structure - Directed Graph

D.4. SIMULATION NODE TYPES

Table D.4: Process Types

Key	Value	Description	Purpose
MECHANIC	mechanic	Basic mechanical or highly predictable process	For very simple, repetitive technical tasks
AUTOMATIC	automatic	Fully automatic execution	To model non-human intervention processes
REPETITIVE	repetitive	Tasks repeated identically	For routine, low-variation tasks
ROUTINE	routine	Standard operating procedures	To capture everyday business operations
STRUCTURED	structured	Well-defined and documented processes	Ensures process clarity and consistency
PROCEDURAL	procedural	Stepwise with clear rules	For rule-driven workflow automation
DIAGNOSTIC	diagnostic	Processes with decision points	Enables error handling and problem solving
ANALYTICAL	analytical	Includes data analysis and reasoning	For data-driven decision processes
SYNTHETIC	synthetic	Combines inputs to create outputs	For creative assembly or integration
ADAPTIVE	adaptive	Adjusts based on environment or feedback	Models dynamic, flexible processes
CREATIVE	creative	Involves innovation and new idea generation	For processes requiring human creativity
EXPLORATORY	exploratory	Open-ended, investigative processes	Supports research and discovery
INNOVATIVE	innovative	Implements new and transformative ideas	Drives competitive advantage and change
TRANSFORMATIVE	transformative	Major change-driving processes	For high-impact strategic transformation

Table D.5: Collaboration Types

Key	Value	Description	Purpose
AD_HOC	adhoc	Unstructured and informal collaboration	Supports spontaneity and quick interaction
INFORMAL	informal	Casual communication without predefined roles	Encourages flexibility and human-centric communication
COORDINATED	coordinated	Participants follow basic sequencing or rules	Helps align minimal structure across actors
SHARED_CONTEXT	shared_context	Common understanding or shared data	Reduces misalignment and increases semantic clarity
PROCESS_DRIVEN	process_driven	Embedded in workflows or formal processes	Enables repeatability and automation
CROSS_FUNCTIONAL	cross_functional	Spanning different roles, teams, or departments	Fosters holistic problem solving
GOAL_ALIGNED	goal_aligned	Participants aligned toward a shared goal	Maximizes productivity and coherence
STRUCTURAL	structural	Organizational structures support and define collaboration	Clarifies accountability and ownership
ROLE_ORIENTED	role_oriented	Collaboration defined by clear role responsibilities	Ensures task specialization and delegation
STRATEGIC	strategic	Aligns with long-term business objectives	Supports sustained, directional collaboration
SYSTEMIC	systemic	Embedded into enterprise systems and architecture	Ensures persistence and integration across the organization
DIGITAL	digital	Enabled by digital platforms and tools	Scales collaboration and improves accessibility
INTERORGANIZATIONAL	interorganizational	Across organizations or ecosystems	Supports partnerships, outsourcing, and supply chain models
MISSION_CRITICAL	mission_critical	Collaboration is vital to business continuity and success	Requires trust, performance, and redundancy

Table D.6: Function Types

Key	Value	Description	Purpose
SIMPLE	simple	Basic, standalone functions	For simple tasks with limited scope
CONDITIONAL	conditional	Functions with decision logic	Supports processes with branching behavior
SEQUENTIAL	sequential	Linear sequence of activities	Models stepwise workflows
PARALLEL	parallel	Concurrent execution of functions	For multitasking and parallel processing
EVENT_DRIVEN	event_driven	Triggered by events	Supports reactive systems
INTEGRATED	integrated	Functions connected across components	Ensures smooth interoperation
ORCHESTRATED	orchestrated	Managed coordination of multiple functions	For complex workflows requiring control
AUTOMATED	automated	Minimal human intervention	Reduces manual effort and errors
MONITORED	monitored	Includes feedback and monitoring	Enables performance and health tracking
OPTIMIZED	optimized	Continuously improved for efficiency	Supports process excellence
CAPABILITY	capability	Represents business or system capabilities	Models reusable competencies
STRATEGIC	strategic	Supports strategic goals	Aligns with organizational vision
CRITICAL	critical	Essential for operation or compliance	Ensures reliability and legal adherence
TRANSFORMATIONAL	transformational	Drives major change	Enables innovation and business transformation

Table D.7: Integration Types

Key	Value	Description	Purpose
NONE	none	No integration	Represents isolated systems
MANUAL	manual	Human-mediated integration	For low-tech or legacy processes
FILE_BASED	file_based	Data exchanged via files	Simple batch data transfers
SCRIPTED	scripted	Custom scripts for integration	Lightweight automation
POINT_TO_POINT	point_to_point	Direct system-to-system links	Simple, limited integration

Key	Value	Description	Purpose
API_BASIC	api_basic	Basic API calls	Enables service communication
API_MANAGED	api_managed	API with governance and throttling	For scalable, secure integrations
SERVICE_ORIENTED	service_oriented	SOA-based integration	Supports modularity and reuse
EVENT_BASED	event_based	Asynchronous event messaging	Enables reactive, loosely coupled systems
MIDDLEWARE	middleware	Integration middleware or ESB	Provides orchestration and mediation
DATA_STREAMING	data_streaming	Real-time data streaming	For time-sensitive data flows
SEMANTIC	semantic	Shared ontologies or vocabularies	Ensures meaning alignment
PLATFORM	platform	Shared integration platforms	For multi-party, multi-application ecosystems
ECOSYSTEM	ecosystem	Open APIs and partner networks	Enables collaboration across organizational boundaries

Table D.8: Service Types

Key	Value	Description	Purpose
NONE	none	No service provided	Placeholder or undefined services
STATIC	static	Fixed service with no interaction	For simple, informational offerings
REQUEST_REPLY	request_reply	Synchronous request/response	Common for web services and APIs
SELF_SERVICE	self_service	User-driven service interaction	Empowers end users
TRANSACTIONAL	transactional	Supports transactions	Ensures data consistency
COMPOSITE	composite	Aggregated service of multiple components	Provides complex business functionality
ASYNC	async	Asynchronous, decoupled service	For scalable and event-driven architectures
DISCOVERABLE	discoverable	Service discoverable dynamically	Enables service registries and dynamic binding
CONTEXTUAL	contextual	Adapts based on context or user	Improves relevance and user experience
STATEFUL	stateful	Maintains session or state across calls	Required for complex workflows
SLA_DRIVEN	sla_driven	Governed by service-level agreements	Guarantees quality and availability
MULTITENANT	multitenant	Supports multiple clients with isolation	Reduces costs via shared infrastructure
PLATFORM	platform	Underlying platform service	Enables ecosystems and extensibility
MISSION_CRITICAL	mission_critical	Critical to business operation	Requires highest availability and resilience

Table D.9: Interaction Types

Key	Value	Description	Purpose
INFORMAL	informal	Casual, unstructured interaction	For flexible communication
DIRECTED	directed	One-way communication	Simplifies control and command
SEQUENTIAL	sequential	Ordered, stepwise interactions	Ensures predictable workflows
PARALLEL	parallel	Concurrent interactions	Supports multitasking
REQUEST_BASED	request_based	Explicit request and response	Common in client-server models
BROADCAST	broadcast	One-to-many communication	Efficient for notifications
NEGOTIATED	negotiated	Interaction with mutual agreement	Supports conflict resolution
COLLABORATIVE	collaborative	Joint, cooperative interactions	Enables teamwork and consensus
STRUCTURED	structured	Formalized interaction patterns	Ensures compliance and standards
RULE_DRIVEN	rule_driven	Controlled by policies and rules	Enforces governance
EVENT_TRIGGERED	event_triggered	Initiated by events	Supports asynchronous workflows
MULTICHANNEL	multichannel	Interaction across multiple channels	Improves accessibility and user experience
ADAPTIVE	adaptive	Adjusts based on context or feedback	Enhances relevance and responsiveness
TRUST_ENFORCED	trust_enforced	Includes trust and security mechanisms	Secures sensitive interactions

Table D.10: Device Types

Key	Value	Description	Purpose
DUMB	dumb	Basic device with minimal logic	Low-cost, simple hardware
SENSOR_ONLY	sensor_only	Device that senses environment	For data collection
ACTUATOR_ONLY	actuator_only	Device that acts on environment	Executes control commands
SMART_BASIC	smart_basic	Simple embedded intelligence	Enables local processing
EDGE_ENABLED	edge_enabled	Performs edge computing tasks	Reduces latency and bandwidth
NETWORKED	networked	Connected to network	Enables remote access and communication
EMBEDDED	embedded	Integrated into other equipment	For specialized functions
VIRTUALIZED	virtualized	Software-defined device	Improves flexibility and resource utilization
AUTONOMOUS	autonomous	Operates independently	For self-managed systems
MULTIROLE	multirole	Performs multiple functions	Reduces hardware complexity
SECURE	secure	Includes security features	Protects against threats
MANAGED	managed	Centrally administered	Simplifies operations and updates
CLUSTERED	clustered	Part of a device group	Enables load balancing and redundancy
MISSION_CRITICAL	mission_critical	Critical to core operation	Requires high availability and fault tolerance

Table D.11: Equipment Types

Key	Value	Description	Purpose
BASIC	basic	Simple physical equipment	For basic infrastructure needs
MECHANICAL	mechanical	Non-electronic, mechanical equipment	For manual or non-digital processes
ELECTRICAL	electrical	Powered electrical equipment	Enables automation
COMPUTING	computing	General purpose computers	Supports application execution
SPECIALIZED	specialized	Dedicated equipment for specific tasks	Optimizes for purpose
NETWORK	network	Equipment supporting communication	Enables connectivity
VIRTUAL	virtual	Emulated equipment	Supports cloud or software-defined infrastructure
ROBOTIC	robotic	Automated mechanical equipment	For industrial automation
MOBILE	mobile	Portable equipment	Increases flexibility
MANAGED	managed	Centrally managed physical resources	Enhances control and maintenance
REDUNDANT	redundant	Duplicate equipment for failover	Improves availability
INTEGRATED	integrated	Combines multiple equipment types	Increases efficiency
ENVIRONMENTAL	environmental	Equipment monitoring environment conditions	Supports operational health
MISSION_CRITICAL	mission_critical	Essential for core business operations	Ensures operational continuity

Table D.12: Event Types

Key	Value	Description	Purpose
PASSIVE	passive	Event that is simply observed, not acted upon	Enables logging or monitoring without response
ISOLATED	isolated	Standalone event with no dependencies	For one-time or disconnected occurrences
MANUAL	manual	Event triggered by human action	Captures user-driven inputs or workflows
OPERATIONAL	operational	Tied to day-to-day business or technical operations	Enables routine operational responses
TIMED	timed	Triggered on schedules or fixed intervals	Supports regular monitoring, maintenance, or batch jobs
TRIGGERED	triggered	Fired in reaction to another specific event	Supports chaining and reactivity
ALIGNED	aligned	Designed to support or reflect business or process alignment	Keeps events coherent with business context
CONDITIONAL	conditional	Only triggered under specific rules or logic	Provides targeted and filtered control flow
INTEGRATED	integrated	Coordinated across systems, layers, or domains	Supports system-wide reactions and handoffs
SEQUENCED	sequenced	Ordered in a deliberate process or event chain	Ensures predictable execution and lifecycle handling
COMPLEX	complex	Composite or aggregate event involving multiple inputs	Enables event-driven architectures
CRITICAL	critical	Must be processed reliably and immediately	Requires fault tolerance and availability
RECOVERABLE	recoverable	Supports rollbacks or compensating actions	Enables resilience and exception management
TRANSFORMATIONAL	transformational	Drives significant change or process evolution	Enables innovation and reengineering

D.5. SIMULATION LIFE-CYCLE

Algorithm 1: Simplified - Hierarchical Sub-Flow Node Execution

Input: $nodeId$

```
1  if nodeId executed then
2  |  return
3  end
4  if nodeId has parent then
5  |  wait until parent is RUNNING;
6  end
7  if nodeId has children then
8  |  Update status to RUNNING;
9  |  foreach child in children(nodeId) do
10 |  |  RunNode(child);
11 |  |  if child status is ERROR then
12 |  |  |  Update nodeId status to ERROR;
13 |  |  |  return
14 |  |  end
15 |  end
16 |  Wait until all children FINISHED;
17 |  Update status to FINISHED;
18 else
19 |  Update status to RUNNING;
20 |  Execute node action;
21 |  Update status accordingly;
22 end
```

Algorithm 2: Simplified - Data Transfer and Execution Binding

Input: $nodeId$

```
1  while exists predecessor p with status ≠ FINISHED do
2  |  yield control;
3  end
4  𝒟 ← ⋃_{p∈Predecessors(nodeId)} FETCHOUTPUT(p);
5  𝒞 ← BIND(𝒟);
6  ℛ ← INVOKE(𝒞);
7  ATOMICUPDATE(nodeId, ℛ, logs, metadata);
8  DISPATCH(nodeId, ℛ)
```

Algorithm 3: Simplified - Error Propagation and Cancellation

Input: *nodeId*

```
if node nodeId status is ERROR then
    foreach successor in Successors(nodeId) do
        Update successor status to CANCELLED;
        PROPAGATEERROR(successor);
    end
end
```

D.6. SIMULATION METRIC OUTPUT

Table D.13: Simulation Metrics Computed from Architectural Simulation Results

Metric Name	Description
BITAI Score	Composite index assessing overall architectural integrity and performance.
Coverage Entropy	Measures the distribution and diversity of coverage across components or layers.
Traceability Ratio	Quantifies the extent to which requirements and design elements are traceable throughout the architecture.
Integration Density	Represents the degree of interconnectedness among components or services.
Layer Coverage Ratio (Business)	Proportional completeness and coverage of the architecture within the business layer.
Layer Coverage Ratio (Technology)	Proportional completeness and coverage of the architecture within the technology layer.
Layer Coverage Ratio (Application)	Proportional completeness and coverage of the architecture within the application layer.
Goal Satisfaction Index	Indicates the degree to which the architecture meets predefined business and technical objectives.
Business Strategy Alignment	Evaluates the congruence between the architecture and the overarching business strategy.
Layered Design Clarity	Assesses the clarity and separation of concerns within layered architectural designs.
Traceability Across Layers	Measures the effectiveness of maintaining traceability links across different architectural layers.
Architecture Flexibility	Captures the ability of the architecture to accommodate changes with minimal rework.
Goal Support Coverage	Reflects how comprehensively the architecture supports specified business goals.
Governance Synchronization	Assesses the alignment and coordination between architectural design and governance frameworks.
System Interoperability	Measures the effectiveness of interactions among system components or subsystems.
Information Sharing Effectiveness	Quantifies the efficiency of information flow across components and layers.
Executive Sponsorship Strength	Indicates the level of leadership support reflected within the architecture.
Reusability of Capabilities	Measures the extent to which architectural components can be reused across different contexts.
Duration (seconds)	Records the runtime duration of each simulation, providing insights into performance and efficiency.

E

APPENDIX E: EA FRAMEWORK SPECIFICATIONS

Table E.1: EA - TOGAF Specification

Aspect	Qualitative Metrics	Quantitative Metrics	Biases & Limitations	Architectural Remarks
Strategy Layer	Traceability of strategic drivers and alignment of business goals with transformation objectives; stakeholder satisfaction with visualization quality	Number of KPIs defined and quantitative success criteria assigned; maturity model targets established	Complexity bias due to abstract modeling; potential overemphasis on strategic drivers leading to neglect of operational details	Emphasizes rigorous linkage between strategic intent and architectural work packages; benefits from modularity in Cloud-Native environments
Business Layer	Completeness of business service definitions and stakeholder influence mapping; process complexity evaluated via actor-role relationships	SLA compliance rates (e.g., >99%); counts of business processes, functions, and roles mapped	Bias towards well-defined processes may obscure emergent or informal workflows; risk of misrepresenting actor influence	Facilitates detailed mapping of business processes to architectural components; complexity metrics support process optimization
Application Layer	Quality of application interface modeling including reusability levels; traceability from business services	Number of application components with full traceability; ratio of reusable interfaces	Possible bias in qualitative reusability assessments; challenge in capturing dynamic application collaborations	Supports flexible componentization and reusability; traceability ensures business alignment
Technology Layer	Modeling of infrastructure resilience and security constraints; system availability and scalability considerations	Server uptime metrics (target >99.9%); response latency < 100ms; redundancy measures quantified	Bias towards high-availability infrastructures may not reflect cost constraints; security model abstractions may overlook novel threats	Critical for mission-critical service support; event-driven facilitates responsiveness and scalability
Impl and Migration Layer	Clarity of deliverables and work packages aligned with strategic outcomes; roadmap visualization quality	Number of work packages linked to outcomes; effort estimations in person-months; gap counts	Risk of underestimating complexity and dependencies; effort estimates prone to bias from optimistic assumptions	Provides structured migration planning; weighting mechanisms assist prioritization and risk mitigation

Table E.2: EA - Zachman Specification

Aspect	Qualitative Metrics	Quantitative Metrics	Biases	Architectural Remarks
Strategy Layer	Traceability of goals, drivers, and outcomes ensures clear rationale; supports cross-cell capability alignment.	Coverage of motivation elements linked to courses of action.	Potential bias toward overemphasis on strategic artifacts limiting operational flexibility.	Enables rigorous motivation-to-action mapping but requires disciplined governance to avoid siloing.
Business Layer	Comprehensive stakeholder perspective separation ensures clarity in role and process definitions.	Number of modeled business actors, roles, and process instances; completeness of “Who”, “What”, “How”, “Where”, “When”.	Risk of inconsistent granularity due to multiple viewpoints causing fragmentation.	Strong separation aids modularity; risks in maintaining consistency across distributed viewpoints.
Application Layer	Alignment of application components to business functions facilitates traceability.	Count of application services and data objects mapped to business elements.	Possible bias toward technical-centric views overshadowing business priorities.	Supports flexible service modeling but may introduce complexity without integration standards.
Technology Layer	Explicit hardware-software mapping enhances deployment clarity.	Number of infrastructure nodes and artifact deployments documented.	May undervalue emergent architectures or dynamic scaling aspects.	Clear artifact execution mapping but needs extension for elastic/cloud environments.
Impl and Migration Layer	Models transformation timelines with deliverables enabling progress tracking.	Count of work packages and migration plateaus defined.	Bias toward planned linear migrations may neglect agile or iterative deployments.	Effective for controlled transformation but less adaptive to continuous change.

Table E.3: EA - FEAF Specification

Aspect	**Qualitative Metrics**	**Quantitative Metrics**	**Biases**	**Architectural Remarks**
Strategy Layer	Policy alignment accuracy; governance rigor aligned to federal mandates	Degree of performance goal attainment aligned with CPIC and PRM metrics	Bias towards federal policy-centric goals can reduce adaptability in non-federal contexts	Strong emphasis on measurable, standardized goals supports cross-agency integration; may impose overhead in highly dynamic environments
Business Layer	Inter-agency business process standardization; organizational role clarity	Number of shared business services and common processes operationalized	Potential bias in over-standardizing unique agency workflows, limiting innovation	Facilitates benchmarking and compliance; supports modularization but risks rigidity in evolving business models
Application Layer	Alignment to federal interoperability standards (e.g., NIEM); collaboration effectiveness	Count of interoperable application interfaces and shared data objects	Bias toward federal standards may limit flexibility with emerging technologies	Encourages reusable services and data standardization; enhances scalability and secure inter-agency exchanges
Technology Layer	Compliance with FedRAMP; use of COTS and reusable services	Number of compliant infrastructure nodes and cloud components deployed	Possible bias in favoring approved commercial solutions, potentially limiting innovation	Promotes security and scalability; reusable components reduce costs but may introduce vendor lock-in

Aspect	**Qualitative Metrics**	**Quantitative Metrics**	**Biases**	**Architectural Remarks**
Impl and Migration Layer	Governance via transformation milestones and gap analysis accuracy	Project-level dependency resolution rates and milestone adherence	Bias toward phased federal program structures may slow agile adaptation	Emphasizes structured migration with clear dependencies; supports incremental modernization but may limit responsiveness

Table E.4: DoDAF Framework Performance and Architectural Assessment

Aspect	Qualitative Metrics	Quantitative Metrics	Bias Considerations	Architectural Remarks
Strategy Layer	Models mission outcomes, capability requirements, risk assessments, and capability evolution plans to ensure strategic alignment.	Measured by completeness and accuracy of mission and risk modeling artifacts.	Potential bias towards defense-centric risk paradigms, limiting applicability in commercial domains.	Supports rigorous top-down traceability, facilitating defense regulation compliance and mission-critical planning.
Business Layer	Defines performers, tasks, operational activities, resource exchanges, and value chains emphasizing mission-critical workflows.	Assessed through task coverage and operational activity completeness metrics.	Risk of overlooking non-military operational models, leading to bias in resource and workflow representations.	Promotes clear task-resource mapping enabling efficient operational analysis and inter-organizational coordination.
Application Layer	Models system functions and logical data flows aligned with operational activities.	Quantified by function coverage ratio and logical flow consistency indices.	Functional bias towards defense application domains may restrict adaptation to civilian IT ecosystems.	Enhances modularity and adaptability, supporting dynamic data flow representation and service interoperability.
Technology Layer	Represents communication networks and infrastructure compliant with Net-Centric Architecture principles.	Network compliance metrics and infrastructure availability statistics.	Infrastructure bias towards defense-grade secure communication standards may limit commercial deployment.	Ensures high security and interoperability standards, essential for mission-critical communications.
Impl and Migration Layer	Defines development timelines, acquisition phases, and compliance gates for system evolution.	Measured by adherence to scheduled milestones and compliance gate passage rates.	May bias toward rigid, waterfall-like development processes unsuitable for agile contexts.	Facilitates controlled system evolution, ensuring compliance and traceability across acquisition cycles.

Table E.5: IAF - Performance and Characteristics

Aspect	Qualitative Metrics	Quantitative Metrics	Biases	Architectural Remarks
Strategy Layer	Clear representation of drivers, goals, and outcome alignment across domains, facilitating strategic coherence and decision traceability.	Coverage of planning horizons and alignment with strategic scenarios measurable via scenario completeness ratios.	Potential bias towards overemphasizing long-term strategic goals, risking underrepresentation of emergent tactical needs.	Structured layering enables governance alignment but may require adaptation in more dynamic environments.
Business Layer	Comprehensive modeling of business functions, policies, and event-driven processes ensuring organizational role clarity.	Quantifiable via policy compliance rate and event-process coverage metrics.	Risk of overlooking informal governance and tacit organizational knowledge.	Supports robust role governance and policy enforcement; suits environments requiring strict process controls.
Application Layer	Emphasizes structured application functions with security and information flow constraints.	Measured by security service coverage and application usage adherence rates.	Security policies might introduce conservative bias, potentially limiting agile innovation.	Focus on security and structured flow supports modular architectures but may increase complexity in loosely coupled systems.

Aspect	Qualitative Metrics	Quantitative Metrics	Biases	Architectural Remarks
Technology Layer	Explicit modeling of infrastructure with sustainability and integration focus.	Quantitative metrics include infrastructure utilization rates and integration success ratios.	Sustainability metrics may bias resource allocation towards short-term gains.	Enhances clarity in infrastructure roles; integration metrics critical for federated deployments.
Impl and Migration Layer	Defines programmatic dependencies with governance checkpoints ensuring migration traceability.	Measured through governance checkpoint adherence and gap closure rates.	Possible bias favoring formalized migration processes, potentially slowing agile transitions.	Facilitates phased implementation; governance checkpoints improve risk mitigation during migrations.

Table E.6: Evaluation of Gartner Framework: Metrics, Biases, and Architectural Suitability

Aspect	Qualitative Metrics	Quantitative Metrics	Biases	Architectural Remarks
Strategy Layer	Alignment of architectural outputs with executive business strategy; stakeholder engagement quality	Measurable business outcome realization rates; throughput of value streams over time	Potential bias toward prioritizing strategic goals over operational detail, possibly obscuring technical debt or implementation challenges	Emphasizes translating strategy into executable initiatives via capabilities; favors clear outcome tracking and strategic transparency
Business Layer	Clarity of business service modeling and decision-support views across domains	Number of business services modeled; coverage of value delivery channels	Risk of oversimplification due to minimizing process-level details, possibly neglecting process optimization opportunities	Focus on high-level service abstraction to enable agile decision making; partitions views for cross-domain clarity
Application Layer	Representation quality of strategic application capabilities and value chain interfaces	Count of application services supporting high-value business functions	Bias toward strategic applications may underrepresent legacy or technical debt in non-strategic apps	Highlights strategic applications as enablers of value chains; supports integration and interface transparency
Technology Layer	Perceived agility and innovation enablement through technology services abstraction	Number of enabling technology services modeled at service abstraction level	Possible neglect of low-level technical constraints and infrastructure realities	Prioritizes agility and innovation over detailed infrastructure modeling; supports abstraction for rapid evolution
Impl and Migration Layer	Effectiveness of milestone tracking and change readiness as perceived by stakeholders	Number of work packages aligned to measurable business outcomes; progress metrics	Bias toward measurable outcomes may overlook qualitative change resistance factors	Supports structured transition planning aligned with strategic objectives; integrates change readiness assessments

Table E.7: Dragon1 Framework - Performance and Architectural Analysis

Aspect	Qualitative Metrics	Quantitative Metrics	Biases	Architectural Remarks
Strategy Layer	Clear visualization of strategic goals and stakeholder motivations enhancing communication.	Number of goals and motivational elements modeled; coverage of stakeholder intent.	Visual abstraction may oversimplify complex strategic dependencies.	Centralized architectures benefit from clear strategic alignment; may lack depth for decentralized systems.
Business Layer	Simplified depiction of key actors and business services improves stakeholder engagement.	Count of business roles and value propositions modeled.	Possible neglect of detailed operational intricacies due to simplification bias.	Suits environments with defined service boundaries; less effective for event-driven or decentralized contexts.

Aspect	Qualitative Metrics	Quantitative Metrics	Biases	Architectural Remarks
Application Layer	Tailored views support stakeholder-specific understanding of application services.	Number of application services and tailored views created.	Potential stakeholder bias toward prioritized views, marginalizing others.	Effective for modular and cloud-centric environments; may require augmentation for event-driven architectures.
Technology Layer	Abstract representation facilitates high-level understanding while avoiding infrastructural detail overload.	Proportion of technology components visualized abstractly.	High-level abstraction risks hiding technical dependencies and risks.	Well suited for cloud platforms; less informative for detailed infrastructure planning.
Impl and Migration Layer	Narrative-driven transformation plans using timelines and work packages enhance clarity.	Number of plateaus and work packages modeled; timeline coverage.	Narrative framing may mask underlying technical complexities.	Best for phased transformation approaches; less agile for iterative or event-driven systems.

F

Appendix F: EA Simulation Models

F.1. Monolith EA Model

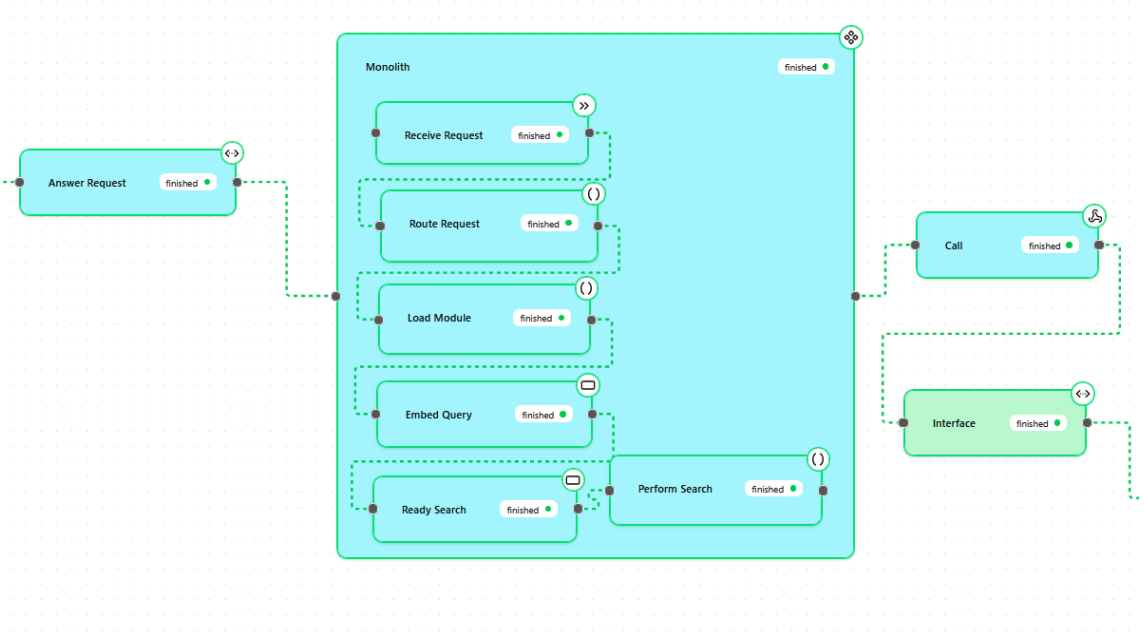


Figure F.1: Monolith Architecture - Application Layer

F.2. Model View Controller EA Model

F.3. Layered EA Model

F.4. Service Oriented EA Model

F.5. Federated EA Model

F.6. Microservices EA Model

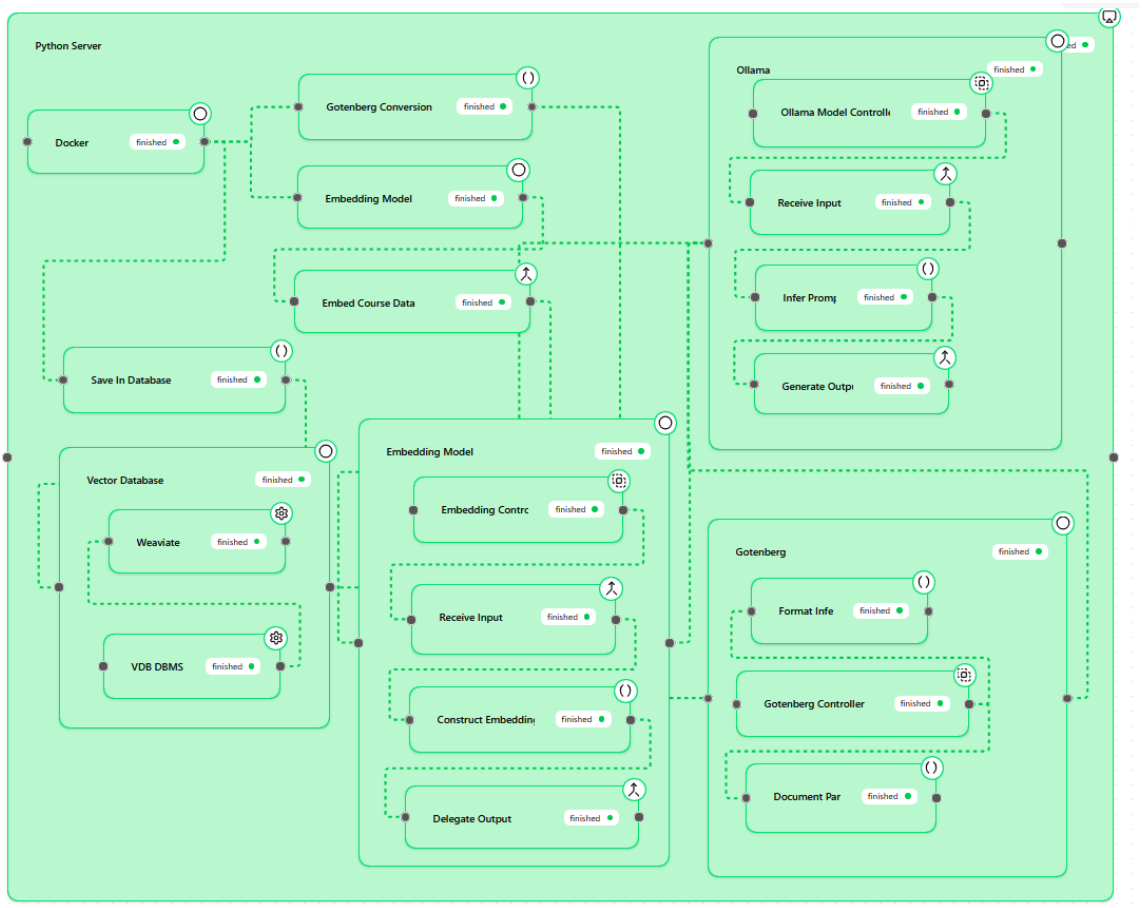


Figure F.2: Monolith Architecture - Technology Layer

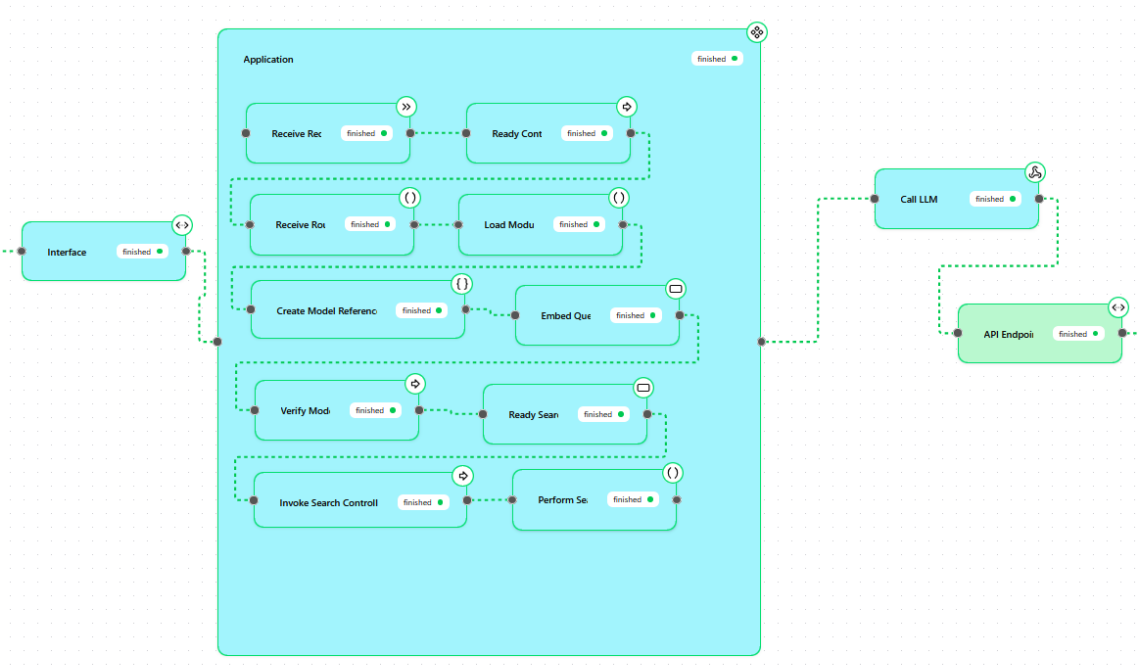


Figure F.3: MVC Architecture - Application Layer

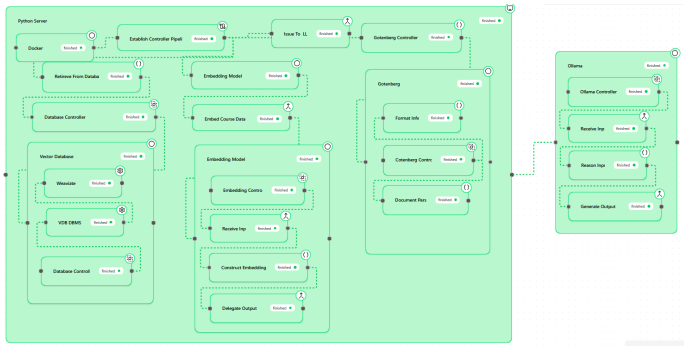


Figure F.4: MVC Architecture - Technology Layer

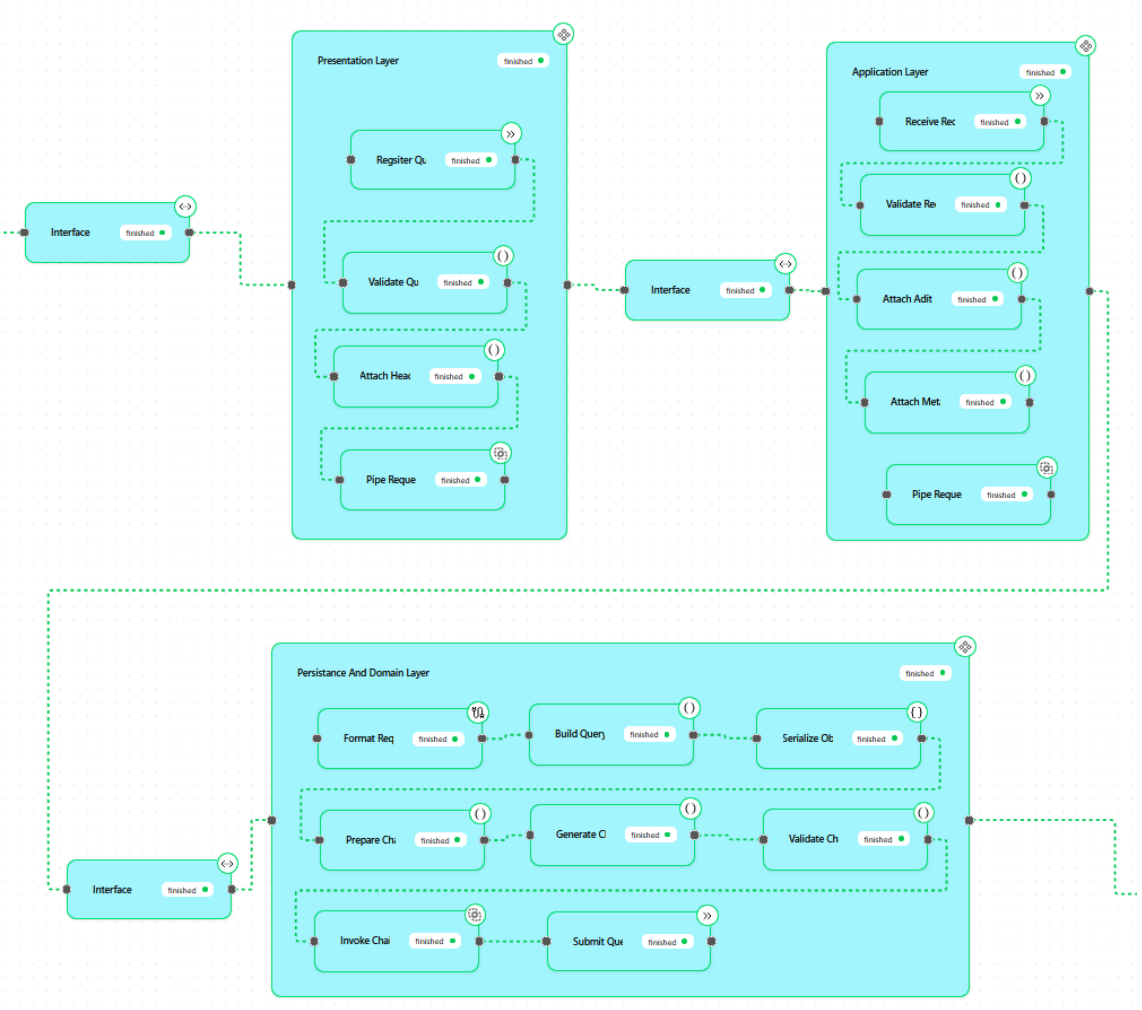


Figure F.5: Layered Architecture - Application Layer

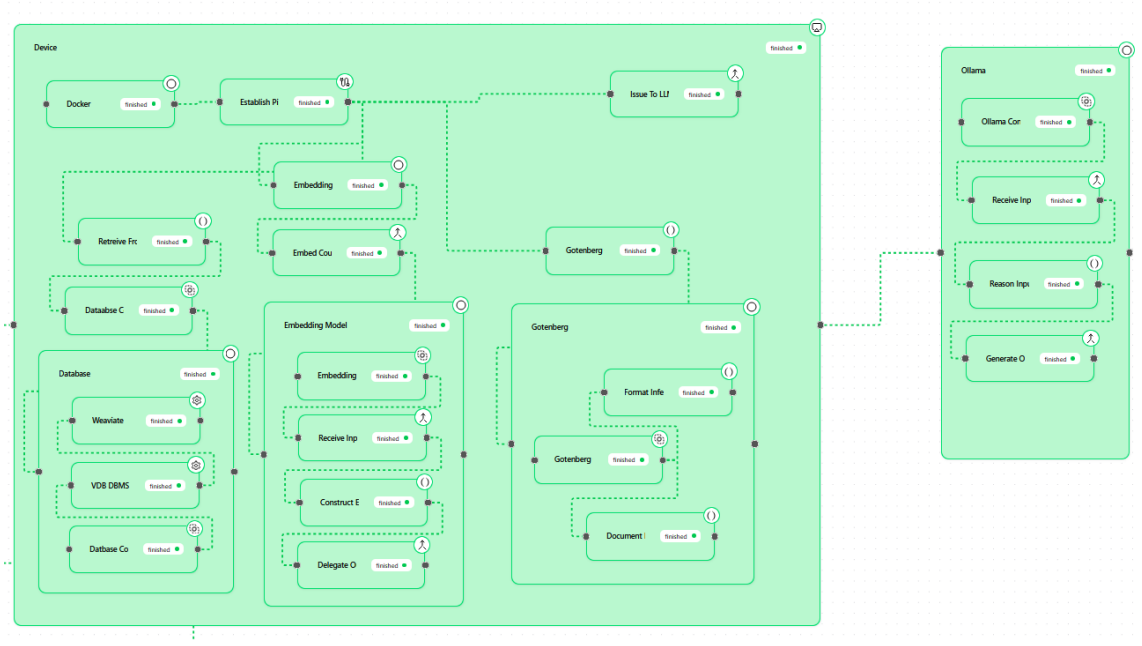


Figure F.6: Layered Architecture - Technology Layer

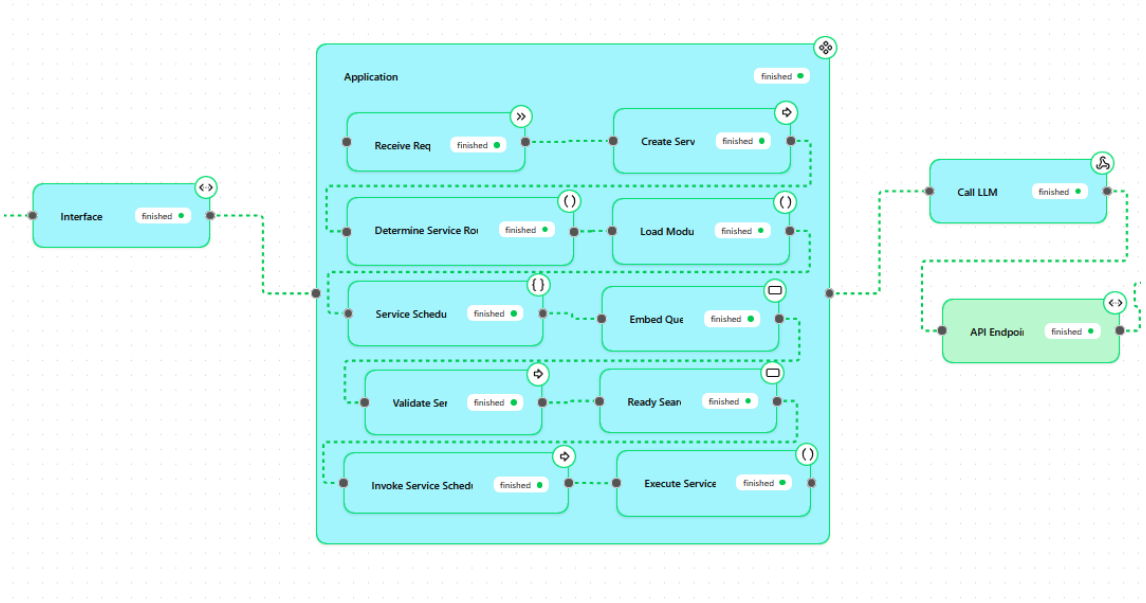


Figure F.7: Service Oriented Architecture - Application Layer

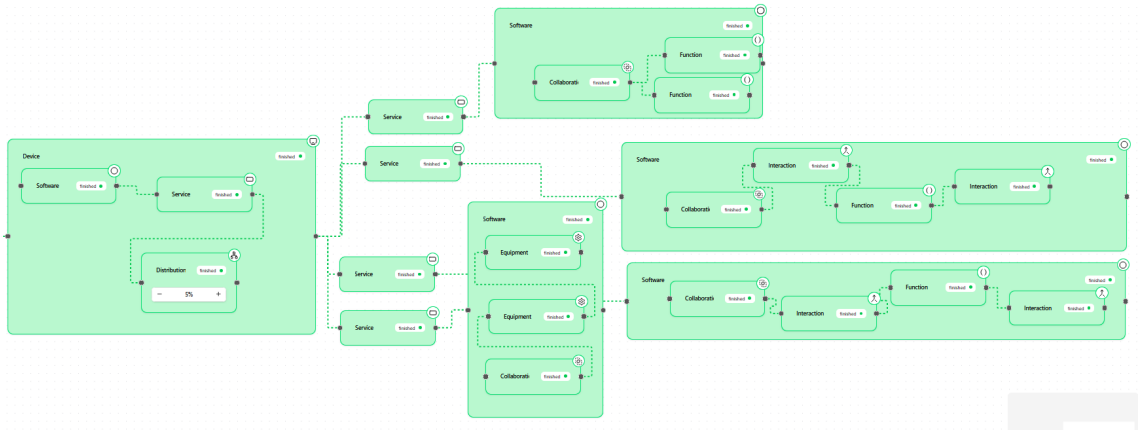


Figure F.8: Service Oriented Architecture - Technology Layer

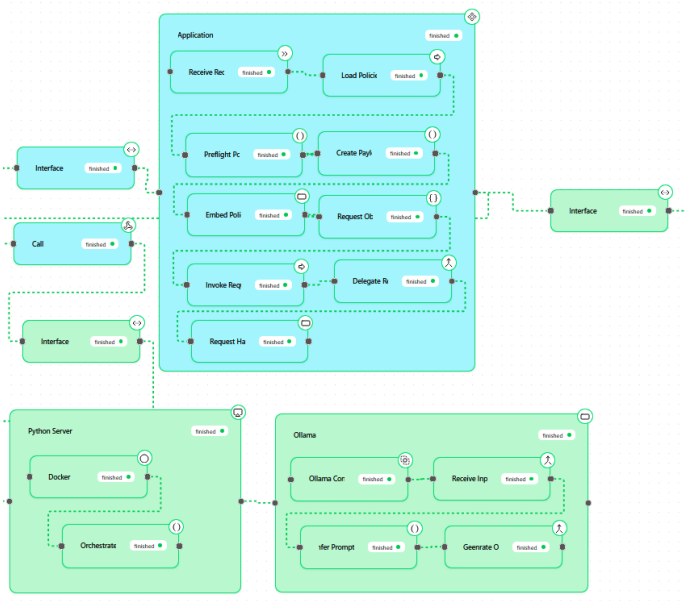


Figure F.9: Federated Architecture - Application Layer

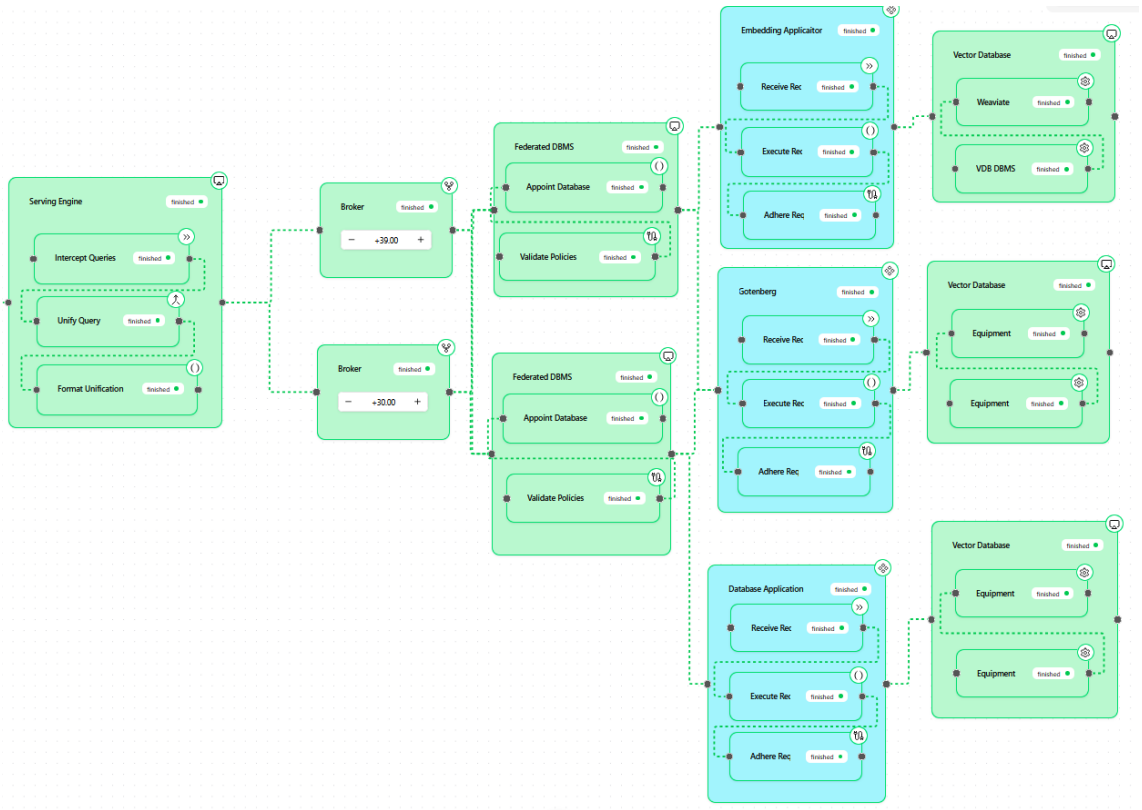


Figure F.10: Federated Architecture - Technology Layer

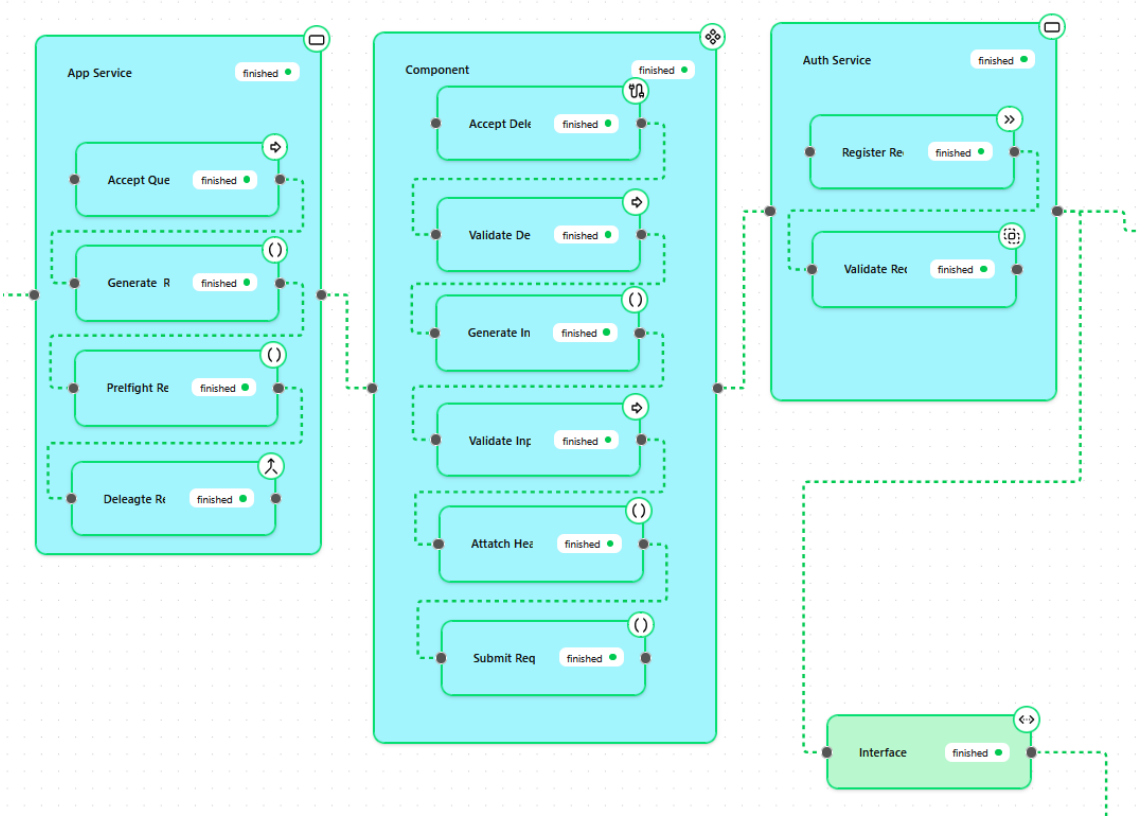


Figure F.11: Microservices Architecture - Application Layer

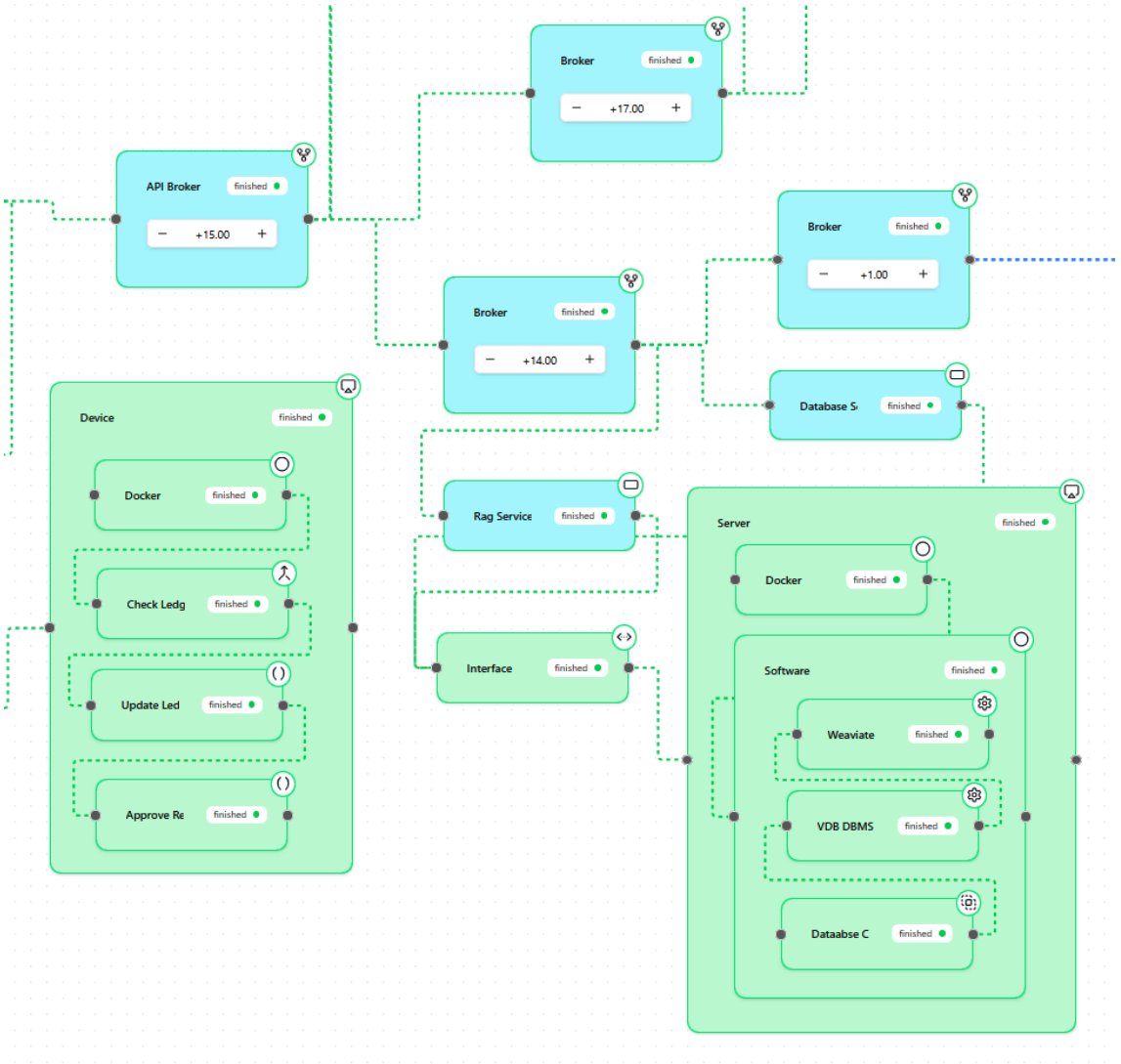


Figure F.12: Microservices Architecture - Technology Layer

G

RESEARCH RESULTS

Table G.1: Combined BM25 RAGAS Scores for LLaMA 3.2:3b and 3.1:7b

Query Nr	Context Recall	Context Precision	Faithfulness	Answer Relevancy
Results for LLaMA 3.2:3b				
0	0.76	0.72	0.67	0.49
1	0.87	0.76	0.68	0.52
2	0.76	0.84	0.70	0.22
3	0.77	0.83	0.72	0.27
4	0.66	0.74	0.68	0.19
5	0.79	0.81	0.67	0.22
6	0.76	0.73	0.58	0.50
7	0.78	0.81	0.69	0.44
8	0.79	0.85	0.72	0.45
9	0.21	0.14	0.34	0.27
10	0.81	0.84	0.71	0.70
11	0.30	0.23	0.44	0.16
Results for LLaMA 3.1:7b				
0	0.81	0.75	0.78	0.66
1	0.90	0.88	0.75	0.70
2	0.79	0.72	0.69	0.39
3	0.81	0.80	0.77	0.49
4	0.73	0.78	0.67	0.35
5	0.72	0.70	0.60	0.41
6	0.66	0.74	0.60	0.70
7	0.81	0.79	0.75	0.62
8	0.80	0.81	0.69	0.63
9	0.61	0.52	0.50	0.45
10	0.83	0.80	0.62	0.69
11	0.53	0.41	0.43	0.31

Table G.2: Cosine Similarity RAGAS Scores for LLaMA 3.2:3b and LLaMA 3.1:7b

Query Nr	Context Recall	Context Precision	Faithfulness	Answer Relevancy
0	0.81	0.58	0.39	0.35
1	0.73	0.67	0.56	0.54
2	0.31	0.16	0.04	0.21
3	0.77	0.73	0.77	0.60
4	0.65	0.63	0.61	0.68
5	0.82	0.82	0.83	0.59
6	0.85	0.80	0.62	0.54
7	0.83	0.83	0.80	0.54
8	0.82	0.79	0.83	0.77
9	0.24	0.16	0.28	0.35
10	0.82	0.82	0.87	0.36
11	0.79	0.78	0.78	0.21
LLaMA 3.1:7b				
0	0.82	0.74	0.56	0.63
1	0.80	0.75	0.67	0.60
2	0.46	0.43	0.32	0.33
3	0.81	0.80	0.74	0.60
4	0.85	0.87	0.65	0.68
5	0.82	0.81	0.62	0.59
6	0.78	0.72	0.63	0.60
7	0.88	0.85	0.72	0.63
8	0.88	0.80	0.68	0.67
9	0.42	0.39	0.46	0.49
10	0.80	0.76	0.68	0.67
11	0.42	0.38	0.38	0.27

Table G.3: Hybrid RAGAS Score for LLaMA 3.2:3b and 3.1:7b

Query Nr	Context Recall	Context Precision	Faithfulness	Answer Relevancy
0	0.89	0.77	0.40	0.35
1	0.87	0.76	0.55	0.54
2	0.40	0.27	0.12	0.10
3	0.77	0.83	0.73	0.60
4	0.71	0.76	0.60	0.63
5	0.76	0.75	0.81	0.60
6	0.76	0.72	0.78	0.33
7	0.82	0.81	0.75	0.54
8	0.85	0.84	0.55	0.60
9	0.21	0.14	0.27	0.34
10	0.82	0.81	0.85	0.36
11	0.67	0.73	0.74	0.21
LLaMA 3.1:7b				
0	0.92	0.83	0.56	0.67
1	0.75	0.75	0.70	0.68
2	0.72	0.64	0.36	0.52
3	0.79	0.84	0.70	0.69
4	0.74	0.78	0.65	0.71
5	0.82	0.87	0.70	0.58
6	0.72	0.75	0.61	0.43
7	0.76	0.82	0.66	0.59
8	0.80	0.78	0.70	0.68
9	0.30	0.36	0.28	0.37
10	0.75	0.82	0.69	0.57

Query Nr	Context Recall	Context Precision	Faithfulness	Answer Relevancy
LLaMA 3.2:3b				
11	0.70	0.75	0.69	0.48

Table G.4: Combined BM25 RAGE Scores for LLaMA 3.2:3b and 3.1:7b

Query Nr	Input Faithfulness	Output Faithfulness	Source Attribution (1 | 0)
LLaMA 3.2:3b			
0	0.61	0.27	1.00
1	0.35	0.50	1.00
2	0.71	0.14	1.00
3	0.53	0.29	1.00
4	0.71	0.16	1.00
5	0.79	0.11	1.00
6	0.52	0.29	1.00
7	0.36	0.44	1.00
8	0.61	0.29	1.00
9	0.53	0.17	1.00
10	0.71	0.08	1.00
11	0.53	0.18	1.00
LLaMA 3.1:7b			
0	0.85	0.38	1.00
1	0.50	0.70	1.00
2	0.73	0.25	1.00
3	0.74	0.41	1.00
4	0.72	0.22	1.00
5	0.81	0.15	1.00
6	0.74	0.41	1.00
7	0.65	0.62	1.00
8	0.85	0.41	1.00
9	0.74	0.24	1.00
10	0.74	0.14	1.00
11	0.71	0.32	1.00

Table G.5: Combined Cosine Similarity RAGE Scores for LLaMA 3.2:3b and LLaMA 3.1:7b

Query Nr	Input Faithfulness	Output Faithfulness	Source Attribution
LLaMA 3.2:3b			
0	0.38	0.37	1.00
1	0.25	0.37	1.00
2	0.32	0.27	1.00
3	0.26	0.24	1.00
4	0.49	0.21	1.00
5	0.33	0.16	1.00
6	0.49	0.24	1.00
7	0.23	0.28	1.00
8	0.49	0.21	1.00
9	0.33	0.10	1.00
10	0.47	0.19	1.00
11	0.29	0.16	1.00
LLaMA 3.1:7b			
0	0.68	0.52	1.00
1	0.45	0.52	1.00
2	0.58	0.49	1.00

Query Nr	Input Faithfulness	Output Faithfulness	Source Attribution
LLaMA 3.2:3b			
3	0.47	0.43	1.00
4	0.69	0.29	1.00
5	0.46	0.22	1.00
6	0.69	0.34	1.00
7	0.41	0.50	1.00
8	0.69	0.29	1.00
9	0.46	0.18	1.00
10	0.66	0.34	1.00
11	0.52	0.22	1.00

Table G.6: Combined Hybrid RAGE Scores for LLaMA 3.2:3b and LLaMA 3.1:7b

Query Nr	Input Faithfulness	Output Faithfulness	Source Attribution
LLaMA 3.2:3b			
0	0.48	0.37	1.00
1	0.38	0.37	1.00
2	0.48	0.27	1.00
3	0.50	0.24	1.00
4	0.50	0.26	1.00
5	0.51	0.16	1.00
6	0.54	0.19	1.00
7	0.41	0.28	1.00
8	0.48	0.21	1.00
9	0.51	0.10	1.00
10	0.49	0.19	1.00
11	0.50	0.16	1.00
LLaMA 3.1:7b			
0	0.77	0.52	1.00
1	0.68	0.52	1.00
2	0.86	0.35	1.00
3	0.70	0.34	1.00
4	0.70	0.37	1.00
5	0.71	0.22	1.00
6	0.86	0.34	1.00
7	0.70	0.50	1.00
8	0.86	0.37	1.00
9	0.38	0.46	1.00
10	0.70	0.34	1.00
11	0.75	0.22	1.00