Leiden University

ICT in Business and the Public Sector

Beyond Technical Optimisation: Design Decisions and Organisational Constraints in Enterprise RAG

Name:	Menno Bruinsma
Student ID:	s2509555
Date:	10-02-2026
1st supervisor:	Christoph Stettina
2nd supervisor:	Bas Kruiswijk
Company supervisor:	Wesley Kruijthof
Company supervisor:	Babette van Dinten

MASTER'S THESIS

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Einsteinweg 55
2333 CC Leiden
The Netherlands

Abstract

Background: Despite rapid adoption, 80% of organisations report no financial returns from their Generative AI (GenAI) initiatives. Retrieval-Augmented Generation (RAG), which enables organisations to ground large language model outputs in their internal knowledge without expensive fine-tuning, has become a common first GenAI use case. While RAG pilots can be deployed quickly, scaling beyond proofs of concept remains challenging in an enterprise environment. Research on successful RAG implementations is limited, as existing literature predominantly focuses on technical optimisation and evaluation.

Aim: To address the limited understanding of why enterprise RAG initiatives succeed or fail, this research aimed to (i) investigate the factors that influence RAG implementations in large organisations (>250 employees) and (ii) synthesise practitioner insights into actionable guidance for enterprise RAG deployment.

Method: We conducted an inductive, reflexive thematic analysis of 14 semi-structured practitioner interviews, with participants representing diverse roles, organisations, and system scopes. Theoretical saturation was reached at interview 8.

Results: Our analysis revealed that enterprise RAG success primarily depends on organisational constraints, particularly data quality, governance maturity, and stakeholder management. Furthermore, practitioners consistently favoured simple RAG systems over more complex ones, revealing a practitioner-literature gap where academic focus on sophisticated architectures misaligns with enterprise needs. We developed a theoretically grounded framework that illustrates how strategic intent, foundational design decisions, operational design decisions, adoption factors, and constraints interact to determine the success of RAG. From this analysis, we derived 11 design principles emphasising organisational readiness over technical sophistication.

Conclusion: As organisations move RAG from proof of concept to production, lasting success requires investment in organisational capabilities rather than pursuing technical sophistication. Our framework and design principles enable organisations to assess readiness, anticipate implementation constraints, and avoid common pitfalls.

Acknowledgements

I would like to express my gratitude towards my university supervisors, C. Stettina and B. Kruiswijk, for their continuous support and feedback. They supported my vision from the start and allowed me to shape my research independently.

Second, I want to thank my company supervisors, W. Kruijthof and B. van Dinten, who, despite being very busy, always made time to discuss my ideas, provide feedback, and challenge me. In addition, I am grateful to my team at Deloitte for welcoming me and allowing me to be part of their group.

Furthermore, I would like to thank all participants for their contributions to this research. Speaking with experts across different roles and organisations was an insightful experience and enabled me to develop a comprehensive understanding of RAG implementations within an enterprise environment.

Finally, I would like to thank my family and friends for their support during this project. I could not have completed this thesis without them.

Contents

Chapter 1

Introduction

With the rise of large language models (LLMs), organisations recognised the power of Generative AI (GenAI). To avoid being outpaced by competitors, organisations are actively pursuing the adoption of GenAI solutions, as reflected in an expected $644 billion total GenAI expenditure in 2025 (Gartner, 2025). GenAI enables the augmentation or partial replacement of a workforce, resulting in a more efficient and lean organisation when implemented effectively. Despite extensive piloting, organisations often fail to move beyond GenAI experiments. An industry report found that while 80% of companies have adopted GenAI, the same percentage of those companies have not experienced financial value creation (McKinsey & Company, 2025). Many of these GenAI solutions fail in their pilot stages due to their inability to deliver clear business value (Challapally et al., 2025). The gap between GenAI capabilities and valuable operational deployment at scale thus remains a substantial challenge.

Retrieval-Augmented Generation (RAG) is one such GenAI solution that organisations are widely adopting (Lewis et al., 2021). RAG allows general-purpose LLMs to access additional data without the need for expensive retraining or fine-tuning. This enables organisations to incorporate their internal data into an LLM, allowing models to answer questions by grounding them in their organisational knowledge. This makes RAG attractive for knowledge management applications, where the retrieval and sharing of information are core capabilities. In practice, organisational chatbots utilising RAG are often among the first GenAI initiatives organisations run, due to their relatively simple, straightforward use case. This has driven rapid adoption of RAG in enterprise settings (Menlo Ventures, 2024), making it one of the most common initial GenAI applications. Consequently, RAG has become an active research area, with numerous papers exploring architectures, optimisation techniques, and evaluation methods (Asai et al., 2023; Y. Gao et al., 2024; S. Zhao et al., 2024; Oche et al., 2025).

1.1 Problem Statement

While RAG pilots and proofs of concept can be deployed within days, organisations face substantial challenges in moving RAG to production. RAG's perceived technical simplicity can mask organisational complexities that emerge during production development. Challenges that appear manageable in controlled pilot environments, such as data, governance, or infrastructure demands, can require substantial effort when deploying RAG within large organisations (Bruckhaus, 2024). This reveals that RAG success does not just depend on technical excellence, but also requires attention to the contextual factors around it.

These challenges and constraints increase the risk of failed GenAI initiatives (Challapally et al., 2025). Stakeholder expectations are set during proofs of concept, where RAG can quickly and visibly deliver results, leaving them unaware of the many hurdles that production-ready RAG must overcome. Development teams may focus on optimising their RAG pipeline, leaving key organisational constraints unaddressed. Therefore, it is critical for practitioners and stakeholders to gain a holistic understanding of RAG enablers and constraints.

From a strategic perspective, RAG is more than a new technical tool; it can fundamentally change how

organisations interact with their own knowledge. Unlike traditional keyword-based enterprise search systems, RAG enables employees to search knowledge bases using natural language. In addition, RAG has the capability to actively reason on retrieved information by utilising the power of LLMs. These characteristics position RAG as strategic infrastructure for organisational knowledge management rather than just a search enhancement tool.

RAG presents distinct challenges in both strategic and technical senses (Bruckhaus, 2024). These challenges are particularly relevant to large organisations, defined as those having more than 250 employees (OECD, 2026). They have diverse data sources, complex governance structures, and distributed users that increase implementation difficulty. Enterprise RAG must integrate knowledge across organisational silos, addressing issues such as access control, data freshness, and governance policies. Moreover, these systems face scalability and cost pressures as they must serve a large number of concurrent users.

The current scientific literature is primarily focused on the technical and optimisation aspects of RAG, proposing new architectures that improve performance by only small margins (Y. Gao et al., 2024; S. Zhao et al., 2024; Singh et al., 2025). However, whether these architectural innovations address the challenges practitioners face in enterprise settings remains unclear. While architectural innovations are valuable for improving future RAG possibilities, this research focus led to a literature gap regarding enterprise implementation contexts. Organisations face numerous design choices during development, which shape usability, performance, and value creation. However, the literature provides no systematic guidance on how RAG design choices should reflect strategic intent or how practitioners can ensure system alignment in an enterprise setting. In addition, there are no clear indicators that can predict successful deployments.

This thesis adopts the Knowledge-Based Dynamic Capabilities theory (KBDC) as the primary theoretical lens to analyse RAG's strategic role (Zollo and Winter, 2002; Denford, 2013). According to the KBDC theory, knowledge serves as the primary driver that enables organisations to sense, seize, and reconfigure resources in response to a rapidly changing environment. RAG can enable specific KBDC capabilities by providing infrastructure for retrieving, accessing, contextualising, and sharing organisational knowledge. However, whether and how organisations realise this potential depends on their internal capabilities and the manner in which they handle challenges and constraints.

1.2 Research Questions

To address the identified practical research gap in enterprise RAG implementations, we state the following research question:

Research Question. *What factors influence Retrieval-Augmented Generation (RAG) implementations in large organisations?*

To address this research question systematically, we developed three guiding questions that structure our analysis. Note that these guiding questions will not be answered explicitly in our Discussion and Conclusion chapters, but serve as an analytical tool throughout our research. We state the following guiding questions:

Guiding Question 1. *What strategic reasons drive RAG adoption?*

Guiding Question 2. *What are key RAG decision points, and what trade-offs arise when organisations adopt RAG?*

Guiding Question 3. *What technical, organisational, and environmental factors constrain RAG design decisions?*

1.3 Research Scope

Our research focuses on RAG implementations in large organisations, defined as those having more than 250 employees (OECD, 2026). These organisations face distinct challenges, such as diverse and siloed data sources, complex governance policies, and a multitude of stakeholders. We study systems at various stages of maturity, including proofs of concept, pilot deployments, and systems in production. In terms

of solution scope, we research both enterprise-wide (horizontal) and domain-limited (vertical) implementations. Horizontal systems serve multiple departments, providing broadly applicable capabilities across the enterprise. In contrast, vertical systems are limited to a single team or department and are tailored to a specific niche of value creation. While vertical solutions may utilise information from multiple departments or data sources, their target users remain limited. Horizontal solutions are typically less specialised, as they aim to serve a large number of users with relatively general information, such as an internal HR chatbot that utilises RAG.

This research views RAG from a knowledge management perspective, specifically for internal question answering (QA). QA is the primary application of RAG, both in research and industry (Y. Gao et al., 2024; McKinsey & Company, 2024; Krishna et al., 2025; Brehme et al., 2025). RAG QA extends traditional keyword-based enterprise search by adding reasoning capabilities and natural language understanding, enabling a more interactive QA process. Within knowledge management, we examine the theoretical potential of RAG through the KBDC lens.

We exclude client-facing RAG solutions such as support chatbots, product recommendations, and client portals. These client-facing applications serve distinct strategic purposes, as they focus on enhancing the customer experience and reducing support costs rather than the management of internal knowledge. While both internal and external-facing RAG systems face similar challenges, developing client-facing applications is more straightforward to scale since they generally require only a small static knowledge base that utilises high-quality documentation. Because this research focuses on RAG as a potential enabler of Knowledge-Based Dynamic Capabilities, we exclude client-facing RAG applications and focus on internal, employee-facing RAG systems for organisational knowledge management.

We do not examine operational practices such as deployment automation, system health monitoring, or maintenance workflows (LLMOps/RAGOps), as these concern system operations rather than strategic design and implementation. However, we include RAG evaluation, as it poses unique challenges not present in standard software systems.

Unless stated otherwise, all references to RAG in this research refer to internal knowledge management systems serving organisational users, whether domain-specific or enterprise-wide. Domain-specific RAGs operate within an enterprise environment but are limited to a department, team, or a specific niche.

1.4 Overview of the Thesis

This thesis is structured as follows:

Chapter 2 establishes the theoretical foundation by examining RAG through the lenses of strategic management and knowledge management. We position RAG as a potential enabler of KBDC capabilities and introduce the technical paradigms that shape implementation possibilities. We also identify key constraints that organisations face when building RAG systems, drawing on the existing literature.

Chapter 3 describes our inductive reflexive thematic analysis methodology applied to 14 expert interviews. We address research quality through the criteria of credibility, dependability, confirmability, and transferability, and reflect on our positionality as researchers.

Chapter 4 presents findings from our thematic analysis, organised around three main themes: strategic motivations for RAG adoption, design decisions, and constraints. We ground these themes in participant quotes that capture their experiences and perspectives.

Chapter 5 synthesises our findings with theoretical frameworks to answer our research question. We present an integrated framework explaining how strategic intent, foundational design decisions, operational design decisions, adoption factors, and constraints dynamically interact. From this synthesis, we derived 11 design principles for enterprise RAG implementations.

Chapter 6 concludes by summarising our key findings, listing both academic and practical contributions, and proposing directions for future research.

Chapter 2

Background and Related Work

This chapter presents an integrative literature review that establishes the theoretical and technical foundations for our research (Torraco, 2005). We synthesised literature on knowledge-based dynamic capabilities, technical RAG literature, and practitioner case studies. We combined information from academic literature (peer-reviewed journals, scholarly books, and conferences), grey literature (white papers and industry reports), and preprints. While grey literature and preprints are not peer-reviewed, they represent the most current RAG knowledge, as peer-reviewed literature often lags behind actual practices. Since RAG evolution occurs rapidly, we determined that these sources were necessary for our research.

Section 2.1 establishes the importance of organisational knowledge, introduces our main strategic lens (Knowledge-Based Dynamic Capabilities), and explains why current traditional knowledge management is insufficient. Section 2.2 outlines how RAG can improve organisational knowledge management and how it relates to our main lens. Section 2.3 provides a detailed explanation of RAG mechanisms and architectures, establishing the required technical understanding. In addition, it explains some important implementation constraints for RAG.

2.1 The Organisational Knowledge Challenge

Large organisations face critical challenges regarding one of their most valuable assets: their organisational knowledge (Nielsen and Michailova, 2007). Knowledge is often scattered across different systems, departments, and countries. Employees struggle to find the information they are looking for, or are unaware that specific knowledge exists within the organisation. This challenge intensifies with organisational growth, with each new employee, system, or process generating new knowledge (Koivisto and Taipalus, 2023).

From a strategic management perspective, knowledge is a sustainable asset to generate competitive advantage. In contrast to physical assets or technologies, organisational knowledge is difficult to imitate and can be continuously improved (Grant, 1996). This section establishes why knowledge matters from a strategic perspective, why traditional knowledge management systems fail to fulfil their task, and identifies system gaps that necessitate new knowledge management approaches.

2.1.1 Why Knowledge Strategically Matters for Organisations

Knowledge represents a foundational element of organisational competitive advantage. The theoretical evolution of how management theory considers knowledge shows the re-conceptualisation of knowledge from a static organisational asset into an organisational capability. Understanding this progression establishes why knowledge management systems and RAG should be a priority in organisational strategy.

The Resource-Based View - Resources as a Competitive Advantage

In management theory, a competitive advantage refers to an organisation's ability to create value to a degree that its competitors currently do not match. Multiple theories on how to create and maintain competitive advantages have been developed over time. One influential perspective in management

theory holds that competitive advantage arises when an organisation possesses resources that meet the VRIN characteristics (Barney, 1991). These VRIN characteristics state that a resource must meet four dimensions. First, a resource must be valuable, meaning it enables the organisation to exploit opportunities. The resource must also be rare, such that competitors do not have easy access to it. Third, the resource is inimitable, meaning it is valuable yet difficult for competitors to imitate. Last, the resource must be non-substitutable, such that no alternative resource can achieve equivalent outcomes. Together, these conditions define a strategic resource.

Since each resource carries different VRIN characteristics, it becomes the organisation's task to identify and leverage them as effectively as possible. This is why a new characteristic was introduced to VRIN, namely the organisation (O). If a resource is to be used effectively, the organisation must be prepared to do so. Over time, the dimensions of inimitability and non-substitutability were often merged, leading to the VRIO framework, which replaced Non-substitutability with Organisation (O).

The VRIO (or VRIN) characteristics are part of the Resource-Based View (RBV) strategic lens. The term RBV was first introduced in 1984 to shift the strategic management paradigm's focus from products (organisational output) to inputs (internal resources) (Wernerfelt, 1984). RBV theory states that organisations primarily gain competitive advantage by utilising their own resources. The theory is rooted in the Ricardian way of thinking, which emphasises the importance of scarce and immobile resources. Similarly, RBV holds that organisational inputs are heterogeneous and imperfectly mobile, meaning they cannot be easily transferred between organisations. As a result, strategic resources cannot simply be purchased, and even when assets are acquired, they rarely lead to sustained competitive advantage (Barney, 1986). This internal RBV perspective contrasts with well-known theories that focus more on the overall external market position, such as Porter's Generic Strategies (Porter, 1980) and Porter's Five Forces (Porter, 1979).

RBV continued to evolve, and a clear distinction was drawn between resources and capabilities (Grant, 1991; Conner, 1991). Resources were seen as elements that the organisation either owns or controls, such as patents, technologies, equipment, and employees. On their own, these resources generally do not create a competitive advantage. Capabilities refer to the ability to utilise and orchestrate a range of resources to create value. These capabilities stem from processes and systems that enable the effective combination of resources. Accordingly, an organisation may own resources that adhere to the VRIO characteristics, but if it lacks the capabilities to combine these resources effectively, it will most likely not lead to a competitive advantage.

However, RBV has been criticised for being too static, as it does not explain how organisations adapt to rapid changes in their markets (Kraaijenbrink et al., 2010). RBV assumes relatively stable markets, i.e., markets that change slowly over time. However, in practice, and especially in current-day technology, markets move quickly, and valuable resources such as data and algorithms can soon become obsolete. The stability assumption of RBV thus limits the explanatory power of the theory in more dynamic environments.

The Dynamic Capabilities View - Adapting to Fast Changing Environments

Following the RBV school of thought that resources lead to a competitive advantage, a new, less static paradigm emerged: the Dynamic Capabilities View (DCV). Dynamic capabilities are described as: "The firm's ability to integrate, build, and reconfigure internal and external competences to address rapidly changing environments." (Teece et al., 1997). Within DCV, a competence is an organisation's ability to perform a set of tasks using a bundle of resources, ultimately achieving a specific outcome. Dynamic capabilities can therefore be understood as the processes that modify, renew, or reconfigure competences, enabling the firm to respond effectively to a constantly changing competitive landscape.

The main underlying dimensions that determine an organisation's degree of dynamic capabilities are path, position and organisational processes (Teece et al., 1997). The path represents the organisation's past experiences and its evolution over its lifespan. The path shapes how organisations respond to opportunities and risks, influencing their decisions about the future. The position reflects an organisation's current assets, such as technology, knowledge, and relationships. They represent the competitive standing and determine which opportunities can be captured. The organisational processes determine how

resources are managed, combined, and reconfigured.

Over time, the foundation of applying dynamic capabilities was also defined. First, opportunities and threats must be identified in both the internal and external environments (sensing). Second, organisations must be able to seize the sensed opportunities and act accordingly (seizing). Last, an organisation must be able to reconfigure its tangible and intangible resources (reconfiguring)(Teece, 2007).

Dynamic capabilities are explicitly about reacting to rapid, continuous changes to the environment, contrasting with the static nature of RBV. It aligns closely with Schumpeter's thinking, in which the driver of competitive advantage lies in innovation. While RBV has laid the foundation, DCV has become a leading strategic lens in management theory (Wang and Ahmed, 2007; Barreto, 2010; Wilden et al., 2016). Within the DCV school of thought, multiple scholars have proposed that knowledge is the most critical dynamic capability, as it supports all three of its application processes (sensing, seizing, and reconfiguring).

Knowledge as a Competitive Advantage

Knowledge is often seen as a valuable strategic resource (Grant, 1996; Spender, 1996; Halawi et al., 2005). It is defined as: "A fluid mix of framed experience, values, contextual information, and expert insight that provides a framework for evaluating and incorporating new experiences and information" (Davenport and Prusak, 2000). In contrast to regular resources such as labour and capital, knowledge can always be extended and, when used properly, can lead to the creation of new goods and services (Grant, 1996). If we compare knowledge against the original VRIN characteristics, it aligns well with all four. It is valuable because it enables organisations to solve problems and exploit opportunities. It is rare since organisation-specific knowledge is difficult for competitors to obtain. It is inimitable because much of it is embedded in employees' experience and expertise. Finally, knowledge is non-substitutable, as no other resource can fully replace its role in the delivery of innovation. Thus, according to RBV, knowledge should be viewed as a strategic resource.

Knowledge can be divided into two categories, namely tacit and explicit (Nonaka and Takeuchi, 1995). Tacit knowledge can be understood as personal knowledge acquired through experience. It is generally challenging to transfer between individuals because it cannot be shared solely through text or vocal interactions. A good example of tacit knowledge is the ability to drive a car, as it can only be learnt via active practice. In contrast, explicit knowledge is more formal and can be shared fully through textual or verbal explanations. An example is learning the meaning of road signs. This does not require practical experience and can be learned from textbooks, instructional handouts, etc.

These two types also represent the knowledge that lies within an organisation. Most organisational knowledge is inherently tacit, as the working experience of its members outweighs the organisation's documented processes, rules, and information assets (Grant, 1996). In addition to the difficulty of sharing tacit knowledge, organisational members are less inclined to share it because of the loss of their own competitive advantage. If they share knowledge that only they possess with their co-workers, they might weaken their added value to the organisation (Haldin-Herrgård, 2000).

As established earlier, knowledge is a strategic resource according to RBV; however, from the DCV lens, it is also a capability. It enables continuous sensing, seizing, and reconfiguring to respond to environmental changes (Zollo and Winter, 2002). Knowledge is constantly expanding through the sharing, learning, and integration of prior knowledge. Both tacit and explicit knowledge contribute to this continuous expansion. Tacit knowledge drives adaptation and innovation, while explicit knowledge enables the transferability of knowledge throughout the entire organisation. As a result, knowledge is embedded in the three dimensions that determine an organisation's dynamic capabilities. It does not just represent what an organisation was and currently is (the path). It also reflects the current assets (the position) and how it can adapt to future changes and challenges (organisational processes).

In addition to DCV, another paradigm emerged from RBV: the Knowledge-Based View (KBV) (Grant, 1996; Spender, 1996). In this view, knowledge is seen as the most important strategic resource of an organisation. KBV posits that an organisation's primary goal is to collect and integrate knowledge. It can be considered as a conceptual bridge between RBV and DCV, although neither paradigm origi-

nally placed knowledge at its core. However, several scholars have argued that, within DCV, knowledge should be regarded as an essential capability for sensing, seizing, and reconfiguring (Eisenhardt and Martin, 2000; Zollo and Winter, 2002). This has resulted in a combination of the Knowledge-Based View with the Dynamic Capabilities View, culminating in Knowledge-Based Dynamic Capabilities (KBDC) theory.

Knowledge-Based Dynamic Capabilities

Building on the RBV, KBV, and DCV theories, KBDC frames knowledge as both a resource and a driver for dynamic capabilities, enabling sensing, seizing, and reconfiguring. In the literature, the precise division of these three activities has often been unclear. For this thesis, we adopt a simplified typology (see Table 2.1) based on Denford (2013). While other conceptualised dimensions exist (Robertson et al., 2023), we chose this typology because of its internal and external characteristics. In the knowledge context, this split represents whether the system uses internal organisational data and/or external data from partners or the internet. The typology provides a framework to analyse which KBDC capabilities are feasible within enterprise environments.

Table 2.1: A simplified KBDC typology, adapted from Denford (2013)

Capability	**Scope**	**Description**
Integrating	Internal	Recognise internal knowledge sources, facilitate transfer, and integrate knowledge to create value.
Replicating	Internal	Identify and apply existing knowledge and processes in other parts of the organisation to reproduce successful outcomes.
Building	Internal	Exploration of new knowledge within the organisation by recombining existing knowledge; involves experimentation and internal knowledge search.
Reconfiguring	Internal	Redeploy existing resources within the organisation to produce new services/products.
Developing	External	Combine organisational and partner knowledge through joint ventures, R&D, and alliances to create new knowledge.
Assimilating	External	Search, acquire, and internalise knowledge from external sources and partnerships.
Synthesising	External	Combine internal and partner knowledge to create a competitive advantage through joint development or marketing.
Imitating	External	Adopt and adapt external knowledge or practices to improve organisational processes and outcomes.

While the RBV, KBV, and DCV provide valuable foundations, this thesis adopts the KBDC perspective as the main analytical lens for examining RAG's strategic role. RAG is inherently a knowledge technology: it retrieves, contextualises, and generates insights from knowledge bases across the organisation. KBDC provides a framework for analysing which capabilities RAG may enable within an organisational context. Additionally, this lens allows us to see the gap between RAG's theoretical potential and actual organisational practices.

2.1.2 Limitations of Traditional Knowledge Management

Organisations recognise the importance of their knowledge, having invested substantial resources in systems to capture, store, and share knowledge (Davenport and Prusak, 2000; Dalkir, 2011). However, these knowledge management systems (KMS) offer limited search performance and possess structural barriers for knowledge access. For example, an engineer might seek information about a specific technical challenge they encountered, which was already solved by a different department years ago. Traditional KMS returns hundreds of technical documents, requiring the engineer to perform hours of manual inspection to find the solution.

The Evolution of Knowledge Management

The management of tacit and explicit knowledge in an organisation falls under the concept of knowledge management (KM). KM is defined as "the capacity to manage information, including gathering knowledge from internal and external sources, transforming it into new strategies or ideas, and implementing and preserving it" (Idrees et al., 2023). Less formally, KM represents the organisation's capability to gather, share, manage, and apply knowledge to drive innovation. Research has shown that effective knowledge management leads to greater innovation within organisations, aligning with the KBDC lens that states knowledge as the primary driver of innovation (Adams and Lamont, 2003). To illustrate how KBDC relates to knowledge management, we present Figure 2.1. The overlapping region of the Venn diagrams represents the concept of KBDC.

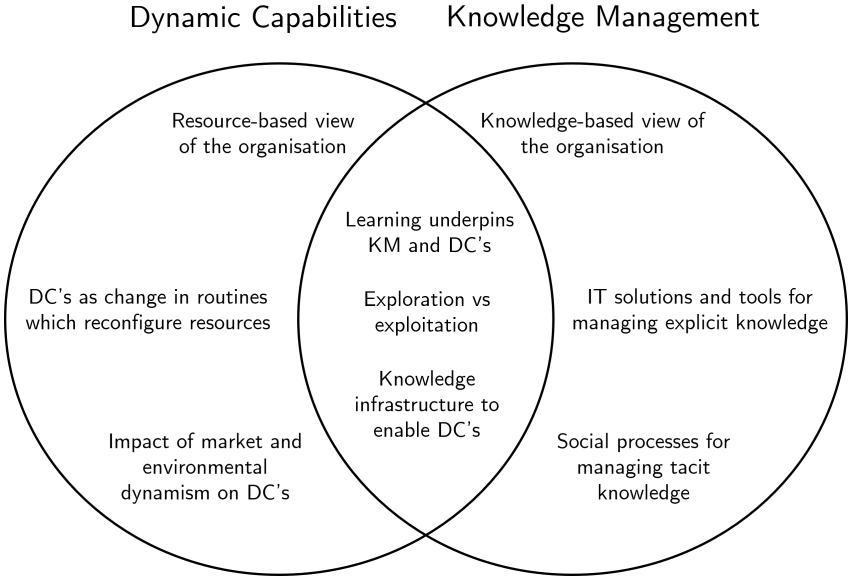

Figure 2.1: The overlap between Dynamic Capabilities and Knowledge Management, adapted from Easterby-Smith and Prieto (2008).

KM evolved over time, often in close relation to new advances in Information Technology (IT). IT systems that support the KM process are known as knowledge management systems (KMS). Early KMS focused primarily on the codification and storage of knowledge (Alavi and Leidner, 2001). Over time, organisations started to recognise knowledge as a strategic resource, as highlighted by the RBV paradigm. This led organisations to design KMS that went beyond mere storage, aiming to facilitate knowledge sharing more effectively (Dalkir, 2011).

KMS typically rely on three basic elements to provide knowledge sharing. First, there needs to be a knowledge base with a well-defined structure, for example, hierarchical folders, that the system can search via a user-provided query. Second, it uses metadata (data that describes data) to provide context for the stored files. Third, the system uses keyword search to quickly search thousands or even millions of documents within the structure. These three elements reflect the capabilities of traditional KMS: quick keyword-based search on a large knowledge base. The goal is to retrieve documents that closely match the given query, placing the responsibility for assessing the retrieved files on the user.

In recent years, KM has entered a new phase (Y. Zhao et al., 2022; Nakash and Bolisani, 2024). New technological advances have expanded the technical possibilities of KMS beyond simple storage and retrieval. With the rise of machine learning and big data, KMS no longer merely stores knowledge; it can also assist in finding, interpreting, and generating insights from it. Solutions such as enterprise search engines, collaborative intranets, and machine-learning–powered recommendation systems demonstrate that IT has become the driving force in how organisations manage their explicit knowledge. However, most organisations still continue to rely on traditional, largely static architectures focused on storage and keyword retrieval (Mohammad et al., 2024).

Challenges in Knowledge Management Systems

Despite these advances in IT, most organisational KMS still rely on traditional architectures that include the fundamental limitations rooted in the three basic elements introduced earlier: knowledge-base structure, metadata, and keyword-based search. These limitations lead to three interconnected architectural problems that systematically prevent traditional KMS from enabling knowledge-based dynamic capabilities.

First, organisations often maintain knowledge bases across multiple systems, utilising heterogeneous data and varying data quality (Nielsen and Michailova, 2007). These are typically separated by different KMS, which reinforces organisational knowledge silos rather than reducing them. While cross-repository or cross-functional KMS exist, they often represent only a small, highly curated subset of organisational knowledge.

Second, the keyword-based search paradigm requires users to formulate queries using nearly the same terminology as in the target file(s) (Xiao et al., 2018). This creates knowledge barriers, especially in large organisations where definitions across departments may vary in practice. Organisations and terminology also evolve over time, so terms may change and come to represent different concepts than they did in the past. While metadata can mitigate such challenges, it does require standardised governance across the entire organisation.

Third, traditional KMS are file retrieval systems, not reasoning systems. They are well-suited for fast keyword-based retrieval but cannot synthesise the information from retrieved documents. Users must manually review and integrate retrieved information, thereby increasing the workload of synthesis.

These three limitations directly hinder traditional KMSs' ability to enable the KBDC capabilities introduced in Section 2.1.1. For example, integrating fails when employees cannot discover knowledge that lies outside their own department due to varying terminology. Replicating capabilities require reasoning to discover previous best practices within the organisation, which cannot be achieved through keyword matching across previous projects. Since traditional KMSs cannot reason, they cannot enable any KBDC capability, as all KBDCs require some form of reasoning.

Traditional KMS also pose a KM paradox: the more knowledge an organisation has, the harder it will be to retrieve and utilise that knowledge. Keyword-based search retrieves documents that appear relevant to the query, but it quickly becomes a needle-in-the-haystack problem for users to find the information they are seeking. Empirical research confirms the poor scaling of KMS within organisations: a comprehensive study found that knowledge management effectiveness follows an inverted "V" pattern, where transitioning from small to large organisations improves knowledge management, but transitioning from large to very large causes effectiveness to decrease despite greater KMS investment (Al Yami et al., 2021).

2.2 RAG as a Knowledge Enabler

This section examines RAG in four ways. First, we briefly introduce the fundamentals of RAG mechanisms. Second, we analyse how RAG addresses the limitations of traditional KMS. Third, we identify which KBDC capabilities RAG might enable in an enterprise environment. Last, we explain what organisations should consider when aligning RAG implementations with their strategic objectives.

2.2.1 RAG Fundamentals

The emergence of Transformers (Vaswani et al., 2017) and language models, such as BERT (Devlin et al., 2019), has led computers to improve their understanding of natural text. With these models, natural text could now be mapped to a vector space via 'tokens', where each vector captures the semantic meaning of its corresponding token. A token can be a single word or sub-word that is mapped to a vector (Mikolov et al., 2013). For example, the word 'profit' could be represented as a vector with 768 dimensions (as in BERT). The distance between vectors in the vector space can then be calculated, for instance, using cosine similarity or the dot product. The distance between two vectors indicates how closely related the tokens' meanings are. For example, the vectors for 'profit' and 'revenue' would be relatively close, reflecting their semantic similarity. This method allows models to capture relationships between tokens in the vector space and forms the basis for natural language understanding.

Despite these advances in text representation, models like BERT remain limited by their fixed training data, as they can only answer questions about information they have seen during training. To incorporate new information, they would need to be fine-tuned or retrained, which is typically an expensive and complicated procedure. RAG directly addresses this constraint by enabling LLMs to access and reason with information outside their original training data, thereby allowing them to utilise new information during inference.

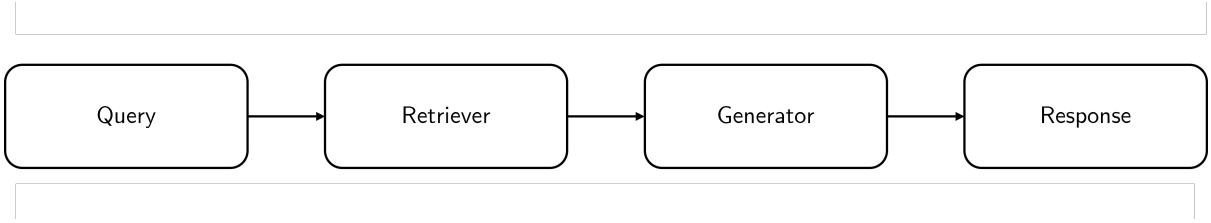

Figure 2.2: The basic elements of a RAG system.

A basic RAG pipeline has four different elements: see Figure 2.2. The query is a question or statement that a user wants an answer to. The query is passed to the retriever, which fetches organisational documents relevant to the query. The query and the retrieved documents are then passed to the generator, which is an LLM instance. The model uses its own internal knowledge, along with the retrieved documents, to generate an answer to the given query. The generated answer is then returned to the user in the output response. This simple process forms the foundation of how LLMs can answer based on added organisational knowledge. More technical details on RAG mechanisms will be provided later in this chapter (Section 2.3).

RAG thus leverages LLM capabilities by integrating them with organisational knowledge bases. This solution design is fundamentally different from traditional KMS: rather than requiring users to know where information resides and what keywords to search, RAG enables employees to ask questions naturally and receive synthesised answers that combine information from multiple sources (Y. Gao et al., 2024).

2.2.2 How RAG Resolves KMS Limitations

RAG directly addresses the limitations of KMS—repository silos, keyword-based search, and the lack of reasoning capabilities—introduced before, see also Table 2.2. The first major limitation within KMS is the typical spread of information across multiple silos (Nielsen and Michailova, 2007). RAG directly addresses this concern by converting all files into vector representations and storing them in a unified vector database. Assuming that all necessary data is converted to vectors, the system can infer variations in terminology and how those terms evolve over time. Even if terminology difference is not captured initially, the system can be improved by providing clear, unified definitions during system operations. Therefore, RAG mitigates silos by removing the need to know where files reside, who created them, or what terminology was used. Users can simply ask in natural language what type of knowledge they are seeking, potentially reducing the time spent on knowledge search compared to traditional KMS.

RAG also directly addresses the limitation of keyword-based search by using semantic similarity. Instead of lexical matching, RAG uses vector representations that capture the semantic meaning of text, where concepts with similar meanings lie close together in the vector space. This enables capabilities such as interpreting synonyms, multilingual reasoning, and jargon understanding. Most importantly, it enables users to describe their problem in natural, detailed language, enabling the system to accurately match the returned content to the specific user problem.

RAG's most critical improvement over traditional KMS is the ability to actively reason on retrieved content. LLMs can interpret a user's query and return a human-like answer, grounded in retrieved content from the knowledge base. Users can typically engage in follow-up interactions, such as asking clarifying or probing questions. The LLM thus adds an interpretative, tacit-like layer to explicit organisational knowledge, a capability that traditional KMS cannot offer. As a result, RAG substantially reduces the burden on users by eliminating the need to manually review documents when trying to find relevant information. This does not mean that RAG output requires no manual checking, but it may substantially reduce the overall workload.

However, RAG also inherits fundamental KMS challenges. Early KM research has shown that "effective knowledge management systems involve far more than just technology, encompassing broad cultural and organisational issues" (Alavi and Leidner, 1999). Challenges include motivating knowledge sharing, developing value metrics for knowledge management, and promoting overall system adoption. These challenges equally apply to RAG. Governance frameworks must enable cross-silo retrieval, data quality determines system usefulness, and user adoption is heavily dependent on AI trust and the correctness of answers. Although RAG offers an upgrade over traditional KMS, it faces the same organisational and cultural barriers that technology alone cannot resolve.

Table 2.2: How RAG addresses traditional KMS limitations.

Limitation	**Traditional KMS**	**RAG**	**Key RAG Benefit**
Knowledge Silos	Multiple repositories with different search systems; users must know where files reside	Unified vector database with semantic search across all sources	Cross-silo knowledge access via natural language queries
Keyword-Based Search	Lexical matching requires near-exact terminology align ment	Semantic similarity via vector representations	Understands synonyms, jargon, multilingual content, and detailed problem descriptions
Lack of Reasoning	File retrieval only; users manually review and synthesise documents	LLM-powered synthesis and interpretation of retrieved content	Provides grounded, human-like answers with follow-up capability

2.2.3 RAG Enablement of Knowledge-Based Dynamic Capabilities

Having established that RAG addresses various KMS challenges, the next step is to examine how RAG potentially enables the KBDC capabilities introduced earlier. Table 2.3 provides a comparison between the theoretical potential of traditional KMS versus RAG. It clearly shows that RAG transforms the theoretical KBDC landscape: capabilities that traditional KMS could only partially or not support at all become theoretically feasible with RAG.

Note that this table reflects potential enablement rather than guaranteed outcomes. Whether RAG enables these capabilities in practice depends on architectural design, knowledge base quality, infrastructure, governance, and organisational readiness, which are examined in Section 2.3.

Table 2.3: Comparison of theoretical KBDC enablement: Traditional KMS versus RAG.

KBDC Capability	**Scope**	**Traditional KMS**	**RAG**
Integrating	Internal	Limited	Yes
Replicating	Internal	Limited	Yes
Building	Internal	No	Yes
Reconfiguring	Internal	No	No
Developing	External	No	No
Assimilating	External	No	Yes
Synthesising	External	No	No
Imitating	External	No	Yes

Internal Knowledge-Based Dynamic Capabilities

RAG has the greatest potential to enable internal KBDC capabilities that utilise internal organisational resources. These capabilities represent the core of what knowledge management systems should deliver, but, as shown in Table 2.3, they are not realised by traditional KMS.

Integrating capabilities require recognising internal knowledge sources across functional boundaries, facilitating their transfer, and combining knowledge to create value. Traditional KMS would require users to know where information resides, who created it, and in what context it was created. RAG eliminates these barriers by providing the ability to retrieve cross-silo information via natural-language queries that work regardless of terminology or repository location. Therefore, RAG shows a clear improvement over traditional KMS for the integrating capability.

Replicating involves identifying existing knowledge across the organisation to reproduce successful practices. Traditional KMS struggled because identifying relevant practices requires semantic understanding of success patterns rather than keyword matching on project names. Users had to manually check for legacy processes and practices from previous projects, thereby placing the entire burden on users to identify conceptually similar situations. RAG can identify conceptually similar situations using its natural language understanding and provide relevant best practices based on prior work. Thus, RAG has the potential to transform isolated successes and best practices into replicable organisational knowledge.

The building capability revolves around the exploration of new knowledge by combining existing knowledge through experimentation and internal knowledge search. Unlike integrating or replicating, which primarily focus on effective but simple data retrieval, building requires advanced multi-hop reasoning: building a chain of related concepts across multiple documents, with the system actively managing what and when to retrieve. This requires RAG systems capable of multi-hop retrieval and reasoning, capabilities which are only available with specific RAG architectures (see Section 2.3.1).

Reconfiguring involves resource reallocation decisions and organisational change processes that extend beyond knowledge retrieval and combination, even though RAG may inform them. Therefore, we do not consider reconfiguring a capability that RAG can support.

External Knowledge-Based Dynamic Capabilities

External KBDC capabilities (developing, assimilating, synthesising, and imitating) utilise data sources that lie outside the organisation, primarily those of partners and publicly available information. While RAG could be applied to these external capabilities, it faces practical governance barriers.

Partner governance constraints make it unlikely that they will grant access to their entire knowledge base; instead, they will likely share only what is necessary to achieve their own goals. Partners will view their knowledge as a strategic resource and thus will be cautious in sharing it (Fassnacht et al., 2023). Therefore, we consider developing and synthesising—the external capabilities that mostly rely on partner data—out of scope for RAG. While it is technically possible to achieve these external KBDC capabilities, governance realities make them impractical for most enterprise implementations. Thus, we consider these two capabilities out of scope for the rest of our research.

Assimilating and imitating can be enabled when RAG systems include internet search and external API access. Assimilating involves searching for, acquiring, and internalising external knowledge; imitating involves adopting and adapting external best practices. Internet-enabled RAG can combine internal documents with academic research, regulatory documentation, and industry information, supporting both capabilities. However, this introduces additional retrieval complexity, challenges with source weighting, and increased security risks, which we examine in Section 2.3.2.

In theory, RAG primarily serves as an enabler of internal KBDC capabilities (integrating, replicating, and building), with the degree of enablement depending on technical and organisational factors. External capabilities (assimilating and imitating) can be partially enabled by internet access, though this introduces additional complexity and governance considerations.

2.2.4 Strategic Alignment in RAG Implementations

Henderson & Venkatraman (1994) conceptualised strategic alignment as the fit between business strategy and IT strategy. Their strategic alignment model consists of four quadrants: (i) business strategy, (ii) IT strategy, (iii) organisational infrastructure & processes, and (iv) IT infrastructure & processes. Their theory posits that strategic alignment occurs when choices in one quadrant are coherent with and reinforce choices in the other quadrants. In the context of RAG, this requires that choices in topics such as architecture, knowledge base, and governance mutually reinforce one another, yielding a coherent RAG system.

RAG systems face implementation uncertainties, as many decisions that shape their capabilities, such as the underlying architecture, must be made when understanding the technological and system consequences is limited. This uncertainty creates epistemic risk; the chance that choices misalign with strategic intent due to a lack of RAG understanding.

This section examines various aspects of RAG and strategic alignment. We first introduce the Technology-Organisation-Environment (TOE) framework, which explains how contextual factors constrain which RAG designs are actually feasible. We then distinguish between foundational and operational RAG decisions: those that determine system capabilities versus those that optimise performance. Third, we examine adoption factors that determine whether organisations successfully integrate RAG.

The Technology-Organisation-Environment Framework

Regardless of strategic clarity, RAG design decisions are inherently constrained by organisational context. The Technology-Organisation-Environment (TOE) framework provides a lens to understand how contextual factors constrain organisational technology adoption (Tornatzky et al., 1990). In the RAG context, TOE explains why RAG designs vary across organisations with similar strategic objectives.

The technological factor in TOE comprises the availability of technology and internal IT capabilities. This examines whether organisations possess the technical capacity to adopt and maintain specific technologies, including factors such as existing infrastructure, technical expertise, and legacy systems.

Organisational context refers to the firm's internal characteristics, such as sector, size, culture, and

resources. This dimension examines how organisational readiness, management support, and internal processes shape technology adoption decisions.

Environmental context refers to the influence of external factors on technology adoption, including industry characteristics, regulatory requirements, and competitive pressure. External constraints such as data protection regulations (e.g., GDPR) can directly shape implementation choices.

The TOE framework thus explains variation in RAG designs across organisations with similar strategic objectives. Two organisations pursuing similar KBDC capabilities might adopt different approaches due to factors such as technological maturity, organisational readiness, or environmental pressures. These contextual differences thus determine which RAG systems are feasible within an organisation.

Foundational versus Operational Design Decisions

RAG systems involve many design decisions, spanning architecture, knowledge base scope, infrastructure, and governance. However, not all design decisions have the same strategic impact on the final system. We propose to examine RAG design decisions along two dimensions: foundational decisions that determine system capabilities and create path dependencies, and operational decisions that optimise performance within established boundaries.

Foundational decisions determine which capabilities a RAG system can support by establishing its boundaries in areas such as architecture, governance, and infrastructure. These decisions create path dependencies: they are made early in development and prove difficult to reverse because changing them requires not only technical rebuilding but also organisational transformation in processes, governance structures, and user workflows. Examples of foundational decisions include:

- RAG paradigm: Choosing which technical paradigm should be used impacts the entire pipeline, as it fundamentally shapes what the system will do. Switching between these requires substantial technical restructuring, as well as organisational readiness for complex governance and training in more advanced system functionalities.
- Knowledge base scope: Determining which domains, data types, and sources to include establishes governance requirements and data quality standards. Domain-specific RAG enables fast deployment but creates governance debt as the system scales. In contrast, an enterprise-wide approach introduces a broader set of stakeholders with distinct data requirements.
- Deployment model: On-premises, cloud, or hybrid deployment affects not just scalability and control, but determines compliance feasibility, establishes vendor dependencies, and shapes which organisational units can access the system. Changing deployment models disrupts established access patterns and necessitates a re-evaluation of security.

Foundational choices must be made during the early development stages, but they require a comprehensive understanding of both technical and contextual aspects, which can only be gained through implementation experience. For example, RAG can integrate many sources into the system, but faces governance maturity challenges as it scales across different silos. Teams can ingest comprehensive knowledge bases, but may underestimate the data-quality standards required for reliable retrieval. This creates the epistemic risk introduced earlier: foundational choices may misalign with strategic intent and organisational readiness due to a lack of RAG understanding. An overview of identified foundational decisions can be found in Appendix A, Table A.1.

Operational decisions optimise performance within the technical and organisational boundaries established by foundational choices. These can be tuned, swapped, or improved during and after deployment without fundamentally altering the system's capabilities and require no organisational change. While operational decisions will impact system quality, they represent tactical optimisations rather than strategic constraints. Examples include:

- Model selection: Swapping between different LLMs of similar capability classes to optimise answer quality, latency or costs.
- Guardrail instructions: The rules passed via the prompt can be iteratively improved based on testing and feedback.

- Query improvement: Adjusting query rewriting, expansion, or routing strategies to enhance retrieval precision.

While technical optimisations matter, a fully optimised system cannot overcome constraints set by foundational decisions. For example, a more expensive model cannot overcome poor data quality, and optimising guardrails cannot replace an actual access control system. Organisations that focus solely on operational decisions will encounter performance ceilings when attempting to scale.

This distinction matters because RAG's technical simplicity hides overarching organisational complexity. RAG prototypes can be built in a couple of days, requiring minimal upfront planning. However, when organisations attempt to scale these prototypes, the impact of foundational choices becomes apparent. Teams often build upon the same architectural foundations and design assumptions used in the prototype (Bjarnason et al., 2023). The governance frameworks, data quality practices, and user adoption patterns established during pilots create path dependencies that may be non-scalable, necessitating substantial changes to the system. Organisations will discover that scaling RAG may require substantial organisational transformation in areas such as governance and data quality.

The Role of Adoption in RAG Success

The foundational and operational RAG decisions determine which KBDC capabilities RAG can support, but the RAG system adoption level determines the actual value the system delivers. Even the most complex and well-performing systems will still fail if people do not incorporate them into their daily workflow. RAG inherits these challenges from traditional KMS systems, in which organisational readiness and user acceptance remain an important hurdle (Alavi and Leidner, 1999). Therefore, understanding the determinants of AI adoption is critical to the successful implementation of RAG systems.

The Task-Technology Fit framework states that technology adoption occurs when system capabilities align with task requirements (Goodhue and Thompson, 1995). In the RAG context, this reflects the clear need for foundational decisions to align with the business case and strategic intent. The work that employees must perform must align with the capabilities of the RAG system. According to this theory, systems optimised for specific use cases, such as basic question answering, will achieve high adoption rates. However, when users expect the same simple system to reason intelligently across multiple documents given complex queries, adoption rates should decline because the system cannot perform this difficult task. In contrast, some systems may be over-engineered when only simple question answering is required. This introduces unnecessary features that make the system less user-friendly, potentially leading to a decline in adoption.

Even when the task-technology fit is satisfactory, adoption still depends on the user's perceptions of the system's usability and value. The Technology Acceptance Model identified two major factors for technology adoption: perceived ease of use and perceived usefulness. Perceived ease of use reflects users' perceived effort to operate the system. Perceived usefulness is the extent to which users believe that a system enhances their job performance (Davis, 1989). Both factors shape adoption outcomes, but are influenced differently by RAG design decisions.

Perceived ease of use is shaped by interface design and system complexity. Many users are already familiar with external GenAI tools such as ChatGPT[1] and Gemini[2], making internal RAG applications feel more cumbersome. They are typically slower, less comprehensive, and less polished overall. Furthermore, advanced KBDC capabilities, such as building, require more sophisticated RAG tooling, which introduces additional latency and system complexity. This creates a design paradox: introducing more advanced KBDC capabilities may lower the adoption rate due to a lower perceived ease of use.

Perceived usefulness is directly tied to KBDC enablement. RAG systems demonstrate their usefulness when they can perform tasks related to the KBDC capabilities. However, the extent to which it can do this depends heavily on the quality, reliability, and relevance of the answers. Systems that cannot deliver answers that users perceive as valuable will thus not be adopted. Therefore, building system trust is essential when deploying RAG applications (Akkiraju et al., 2024). Perhaps the biggest challenge of RAG is the risk of hallucinations (Zhou and Lu, 2024). A hallucination is a plausible but faulty answer

[1] https://chatgpt.com/
[2] https://gemini.google.com/

generated by an LLM in response to a given query. If users receive made-up answers, they are quick to discard the tool and revert to manual search. Organisations can partly mitigate hallucinations by implementing citations, human feedback, and faithfulness metrics, but this adds extra complexity to the system.

Successful RAG adoption also requires behavioural change (Ettinger, 2025). Employees need to begin using RAG applications in their daily workflows, querying AI systems rather than performing manual document searches or asking co-workers. Such changes may face resistance from experienced individuals, as they believe they understand both the context and content better than AI (Zirar et al., 2023). Others may lack understanding or familiarity with GenAI, requiring additional training and guidelines.

To overcome this resistance, organisations must be transparent and open about system capabilities by demonstrating clear value through use cases and training. However, change management alone cannot overcome fundamental task-technology misfit or systems that lack usefulness. Therefore, foundational choices must align with strategic intent, and operational decisions must align with requirements such as latency and ease of use.

In this research, we measure RAG success primarily through adoption rates within the intended user population. High adoption among target users reflects a strong technology-task fit, perceived ease of use, and perceived usefulness. As such, a domain-specific RAG system achieving 70% adoption among a 30-person team demonstrates greater success than an enterprise-wide system used by only 10% of thousands of potential users.

Technical performance metrics, such as answer correctness, retrieval accuracy, and response time, all influence the final adoption rate. However, a system that performs well on technical metrics but shows no adoption should not be considered a success, as potential users do not perceive its usefulness. Therefore, organisations should not rely solely on technical metrics to measure success, but rather on the value they deliver in practice.

Having established the theoretical foundations, we can now examine how these models interact. Each introduced framework addresses a distinct aspect, together forming an integrated system. Figure 2.3 shows how these frameworks connect with each other in a RAG context.

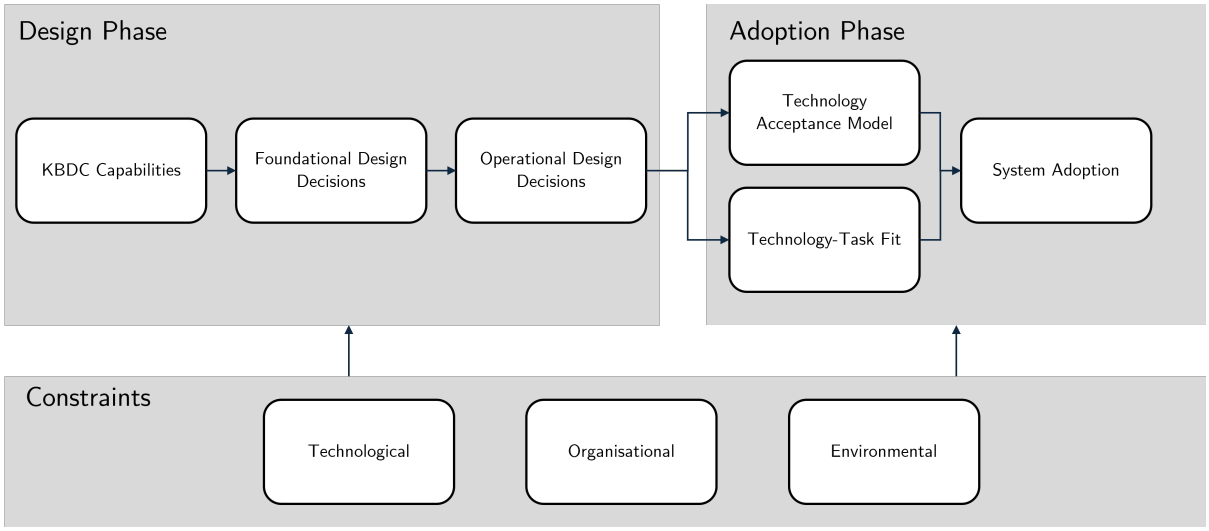

Figure 2.3: Connections between the introduced concepts and frameworks in the RAG context.

2.3 RAG Architectures and Implementation Constraints

The previous sections established the advantages of RAG for knowledge management and strategic alignment in RAG. This section examines the technical mechanisms that shape RAG systems. We will examine multiple interconnected aspects, including the system architecture, knowledge bases, infrastructure, and governance.

2.3.1 The RAG System Paradigms

Although RAG was first conceptualised in 2020, numerous iterations have already followed. Scholars have divided the evolution of RAG into multiple technical paradigms that explain how data is retrieved, generated, and returned (Y. Gao et al., 2024; Singh et al., 2025). There are three main paradigms: Naive RAG, Advanced RAG, and Agentic RAG. We recognise that the paradigms are limited to architecture alone, as they do not consider challenges such as governance, resource constraints, and latency. Technical understanding of RAG serves two purposes for our research. First, the technical paradigm is a primary determinant of system capabilities, thereby directly linking strategy and technical decisions. Second, technical understanding reveals trade-offs and design decisions that practitioners face during RAG implementations. In this section, we discuss query-complexity levels and explain the technical mechanisms of the three RAG paradigms.

Query Complexity Levels

When designing RAG applications, the complexity of the queries which they are expected to handle is a key consideration and can be categorised in various ways. These categories differ in the complexity of the query and the depth of data interaction that is required. In our research, we follow the four query-complexity levels proposed by Zhao (2024).

- Level 1 Explicit Facts: These queries ask about information that can be directly extracted from the knowledge base without the need for extra reasoning. An example question would be: *What are the operating hours for our Amsterdam office?*
- Level 2 Implicit Facts: Implicit facts require some reasoning to answer the question. The answer is not immediately obvious and may be spread across multiple sources. An example question would be: *What training courses would a new data analyst need to complete before accessing customer databases?*
- Level 3 Interpretable Rationales: In this level, RAG is not just able to answer based on implicit rationale, but also able to follow predetermined rationales that are documented in the knowledge space. An example is a decision tree for handling an IT ticket. The system should be able to follow pre-determined workflows, transcending the simple retrieval steps in levels 1 and 2. An example question could be: *What steps should be followed when onboarding a new employee according to the HR knowledge base?*
- Level 4 Hidden Rationales: The hardest level to answer revolves around so-called hidden rationales. In this level, RAG must search for rationales that are not explicitly mentioned in the documents. These rationales are often tacit and pose a substantial challenge to extract. An example question would be: *Three different teams have successfully reduced customer churn last year. What common approaches did they use?*

Not every RAG solution needs to reason at all four levels. Simple question-answering solutions would require only levels 1 and 2. Organisations need to specify in advance what they aim to achieve by implementing RAG, and each design decision that follows should reflect this original goal. This does not mean that the scope of the solution cannot change. If the goal changes over time, the design choices should change accordingly.

These levels not only guide the design process but also shape the role RAG will play within the organisation. The degree to which the RAG system can fulfil satisfactory answers is inherently coupled to the types of queries it can answer. At its simplest, RAG can be a search tool that operates on a small knowledge base. On the more advanced side, it can serve as an orchestration tool that integrates scattered sources to deliver complex answers. It may be tempting for management to seek as many features as possible, but each step up in complexity also entails more difficult design choices, additional

governance issues, and ultimately higher costs. That is why RAG needs a clear strategic perspective so that the design aligns with what the organisation actually wants to achieve.

Architecture of Naive RAG

While we briefly discussed the basics of RAG in Section 2.2.1, this section will go into the mechanics of how RAG works. We use Figure 2.4 to explain how a basic naive RAG solution allows an LLM to interpret organisational data. In this example, the system can only utilise unstructured textual data. Naive RAG is considered to be the foundational paradigm of RAG. Its approach to generating answers based on retrieved relevant information is relatively straightforward and simplistic. It uses simple distance calculations, such as cosine similarity, to retrieve the most relevant documents. In practice, these RAG solutions can only address Level 1 queries, and even then, performance can be minimal, depending on the use case.

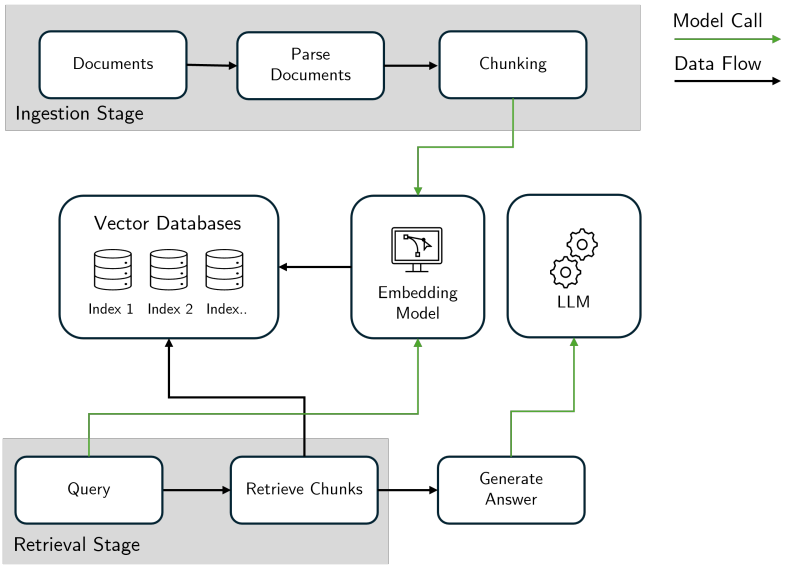

Figure 2.4: The architecture of a Naive RAG system.

The first step to enable an LLM to access organisational textual data is to chunk documents into smaller parts. Each chunk contains a limited set of content from the original document, for example, 400 words. Each chunk is then passed through an embedding model that maps its content to a corresponding vector. The resulting vectors can capture the semantic meaning of the original chunk. The calculated vectors and their corresponding chunks are saved to a vector database. A vector database is a database optimised for storing and retrieving vector data. Thus, a document is represented as a collection of embedded chunks within the vector database, enabling semantic search and retrieval.

When a user prompts a RAG system with a query, the query must be vectorised by the same embedding model that was used to vectorise the chunks. Using the same model, the query will be mapped to the same vector space as the chunk vectors. If another model were used, it would be impossible to accurately compare the query with the chunks in the vector database. Next, retrieval is performed by computing distances between the vectorised query and the vectorised chunks. The retriever then selects the top-k similar chunks from the knowledge base.

The textual content of the top-k chunks, along with the original query, is then fed to an LLM, either an open-source model or a commercial one. The LLM call is accompanied by some simple instructions via the system prompt, such as:

> *You are a helpful assistant who answers questions.*
> *You have been asked the following question:* ***Query****.*
> *Use the following retrieved content to answer this question:* ***Top-k retrieved chunks****.*
> *If the information is not in the context that you are provided with, state that you do not know the answer.*

Upon completion of inference, the model returns the answer to the user as text. While true Naive RAG is rarely deployed in large enterprise settings, an understanding of its basic architecture is required to comprehend the enhancements introduced in advanced and agentic paradigms.

Architecture of Advanced RAG

As Naive RAG evolved, particularly in retrieval steps, it became Advanced RAG. For example, queries were preprocessed to improve retrieval performance. The retrieved content was subsequently post-processed by re-ranking the top-k results. These additional steps led to improved overall RAG performance. This also led to some complications; for example, the more advanced your pipeline is, the higher the latency. There is no real limit on how complicated the pipeline can be to be considered part of the advanced paradigm, except that its execution process remains deterministic. Advanced RAG follows a predetermined path; it has no autonomy to choose its own course. However, it can interact with external sources, such as APIs, if this fits within pre-built capabilities. Advanced RAG would generally be able to answer level-1 and level-2 questions. Level-3 could be possible if the predetermined paths allow access to written rationales.

We sketched an Advanced RAG solution in Figure 2.5, which shows some frequently deployed additions to a RAG pipeline. Many diagrams of Advanced RAG exist, but we chose to modify the diagram proposed by Akkiraju et al. (2024) because it is the most accurate operational representation of the paradigm. Note that this implementation does not accurately represent a specific operational RAG solution; instead, it illustrates an example pipeline and its common elements. In this section, we provide a detailed explanation of this end-to-end Advanced RAG pipeline, including its individual steps.

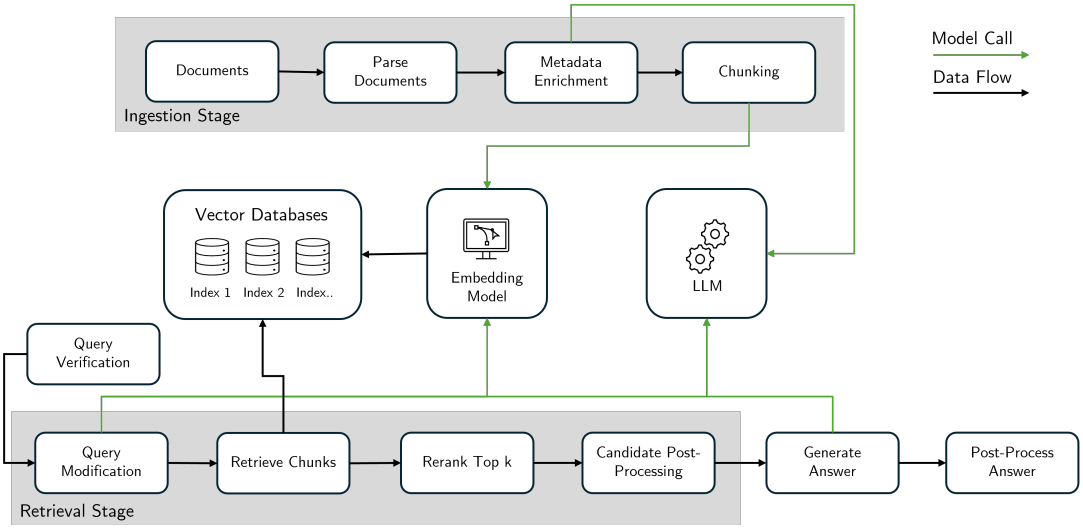

Figure 2.5: An overview of an advanced RAG solution, adapted from Akkiraju et al. (2024)

Ingestion Stage

In Naive RAG, documents were only textual and generally easy to ingest. In practice, organisations have a wide variety of data sources, such as images, audio, and video. Ideally, RAG would be able to ingest all these types so that they can be retrieved and reasoned on. The capability to handle multiple data types is referred to as multi-modal RAG and constitutes an entire research topic in itself. We will not go into technical details of how to build a multi-modal RAG, but achieving such a solution is recognised as a substantial challenge in the literature (Y. Gao et al., 2024; Oche et al., 2025).

The most frequently used capability of RAG is the ingestion of textual data, such as PDFs, Word files, HTML files, and tables (Brehme et al., 2025). First, the documents are parsed and transformed into practical formats for the remainder of the RAG pipeline. After parsing, the documents are enriched with metadata. Metadata is information that describes data, such as a file or document. It adds elements such as labels, timestamps, tags, or context that help explain what the original document contains. It makes data easier to find without altering the actual content. The added metadata is

later used to improve retrieval, generation, and post-processing. Metadata enrichment can be performed using small LLMs, which are inexpensive and fast. Metadata can also be added manually if required, for example, for key documents where metadata must be perfect. While metadata enrichment can be tedious work, it has a positive impact on RAG performance (Akkiraju et al., 2024; Balaguer et al., 2024).

After metadata enrichment, the parsed document needs to be chunked (split). Different data types require tailored chunking strategies. For example, while paragraph-level splitting can be effective for raw text, tabular data requires a different approach. The manner in which documents are chunked is one of the most significant factors that influences RAG performance (Akkiraju et al., 2024; Hasan et al., 2025). If the resulting chunks are too small, the model might not be able to understand their context during retrieval. If they are too large, unnecessary noise is introduced, potentially leading to the retrieval of incorrect chunks. The noise in incorrectly retrieved chunks can also confuse the LLM, resulting in faulty generated answers. The chunking strategy is a crucial component of RAG performance and is heavily dependent on the data files ingested. The ideal splitting strategy is often determined by trial and error and cannot be predetermined. The optimal strategy can also change in response to new documents being ingested into the system.

Embeddings
Another key step of RAG is the embedding process. For example, the embedding model must accurately map domain-specific jargon to a vector space, which can be challenging for a model trained on general data. Organisations might also have data that is in different languages, for example, Dutch and English. A multilingual embedding model could be used, but it is less effective than language-specific (monolingual) models, especially for low-resource languages. Ultimately, embedding performance is dependent on the language and the desired task (Ulčar et al., 2026).

To improve embedding performance, organisations can opt to fine-tune an embedding model on their own datasets (Roychowdhury et al., 2025). This requires additional development time and may only be beneficial if existing models cannot properly integrate their data. Challenges such as these need to be considered in advance; if embeddings cannot capture the intended meaning, the RAG system will not be able to retrieve the correct chunks.

Vector Database
Once the chunks are embedded, they are stored in a vector database specifically designed for fast similarity searches. Unlike standard databases, vector databases enable efficient retrieval of the most semantically relevant chunks from millions of embeddings. This fast retrieval is enabled by the indexing strategy utilised by the vector database. There are multiple indexing strategies, such as HNSW, PQ, and IVFADC, which determine indexing and retrieval speed (Bruckhaus, 2024). Advanced vector databases also support metadata filtering, enabling queries to return only chunks with specific characteristics without scanning millions of vectors.

Alternative retrieval architectures, such as graph-based approaches, represent knowledge as entity-relationship networks rather than embeddings (Edge et al., 2025). The choice between vector-based and graph-based retrieval is discussed in more detail in Section 2.3.3. Note that RAG also offers the option to utilise internet searches, API calls, or other database types. We chose to visualise only vector databases, as this is the most commonly used data source in practice.

Query Verification
The first step when a user submits a query is to check whether it is potentially adversarial or manipulative (Packowski et al., 2024). Adversarial queries can be intentionally vague or contain malicious content to trick the system into returning sensitive information from the knowledge base. Various methods exist for detecting malicious queries, such as applying small LLMs or standard machine learning models to quantify query intent (Guo et al., 2025).

Query Modification
A key distinction between Advanced and Naive RAG is the additional steps added during the retrieval stage. First, the query is modified using multiple methods often used in conjunction to enhance the received query (Y. Gao et al., 2024; Zhu et al., 2024). The listed steps are not necessarily sequential or always implemented; they represent commonly used techniques.

Query Rewriting
User queries may be vague, incomplete, or contain spelling errors. These imperfections can be corrected by rewriting the query; however, this risks losing the original query's intent. Query rewriting can also involve paraphrasing or expanding abbreviations to make domain-specific terminology understandable for the retriever. In practice, this step is typically performed by a small, fast LLM that can process queries at low cost. Overall, this step serves as the first line of refinement by smoothing out user input.

Query Expansion
In addition to rewriting, queries are often expanded to include semantically similar terms, especially when a large amount of domain-specific language is used. Queries are also expanded by generating new queries with a similar meaning to the original. Lastly, queries are sometimes split into smaller sub-queries, especially when handling level-3 and level-4 complexity-level prompts. These expansions help capture the user's intent and improve the likelihood of retrieving relevant chunks. However, generating new queries may also lead to the original intent being lost. Expansion can be performed by generating multiple variations of the original query using an LLM or by using embedding models to find related keywords.

Metadata Tagging
Advanced systems can also assign metadata to the query itself, such as topic, date, or domain tags. During retrieval, this metadata is compared with the metadata stored alongside chunks in the vector database, allowing the system to filter or prioritise results based on relevance criteria that extend beyond similarity calculations (Poliakov and Shvai, 2024). This approach ensures that returned chunks not only match semantically but also align with contextual factors important to the query.

Expected Answer Generation
Another strategy is to generate an expected answer from the LLM before any retrieval occurs. The generated expected answer serves as a guide, highlighting the type of information that the system should focus on during retrieval. As a result, the retrieval process becomes more precise, reducing the probability of returning irrelevant chunks (L. Gao et al., 2022).

Retrieve Chunks
After query modification, the new query is embedded with the same embedding model as was used to embed the data in the vector database. Once embedded, chunk retrieval can begin, setting the stage for processes such as query routing, multi-hop retrieval, and eventual reranking.

Query Routing
The retrieval complexity varies depending on the scope and requirements of the solution. In large-scale RAG, there must be rules or heuristics to guide the retriever's route, since searching across millions of vectors in separate vector indexes would be too computationally expensive. Indexes are often organised into categories, making their contents easy to quantify. The process of directing systems towards specific data sources is known as query routing (Y. Gao et al., 2024) and has a substantial impact on overall retrieval performance. The earlier-mentioned metadata is a tool for routing queries to promising locations, thereby speeding up retrieval.

Multi-hop Retrieval
Beyond query routing, more advanced systems may utilise multi-hop retrieval. In this method, the system iteratively retrieves context until it finds that the retrieved information contains sufficient evidence to generate a final answer. In advanced RAG, the number of hops is often predetermined and terminates after n retrieval rounds. Multi-hop is required to answer more complex query levels (levels two and above) (S. Zhao et al., 2024). It iteratively refines the query using retrieved context and uses the resulting context for the next search round. Although this method still employs a predetermined static path, it introduces limited autonomy, depending on the multi-hop method used (Asai et al., 2023). Multi-hop retrieval can be seen as a bridge between Advanced and Agentic RAG, introducing complex reasoning to the static pipeline of Advanced RAG. We will explain multi-hop retrieval in more detail when the Agentic RAG architecture is introduced.

Rerank Top-k
After retrieval, the top-k relevant chunks are returned from the vector database. These top-k chunks are the most similar to the modified query under a chosen distance metric, such as cosine similarity. In practice, the top-k results from the initial similarity search are not always optimally ordered, as high similarity scores can still include semantically close but contextually irrelevant chunks. For this reason, the returned chunks are often reranked using a weighted calculation that incorporates multiple variables in addition to the previously computed distance. Commonly introduced variables include keyword-based relevance (such as BM25) and a metadata alignment score. The combination of semantic and keyword models is referred to as hybrid scoring (Singh et al., 2025). The keyword models ensure that important terms are captured, while the semantic model captures overall context similarity. Fine-tuned reranker models have also emerged, improving reranking performance at the cost of increased computational cost (An et al., 2025). The goal of reranking is to identify the n most relevant chunks from the original k, thereby improving the foundation for accurate answer generation.

Candidate Post-processing
After the top-n chunks have been retrieved, several candidate post-processing strategies can be applied to refine the input before generation. These strategies are not strictly sequential; they represent optional choices depending on system requirements and domain constraints.

Sensitive Information Detection
A crucial step is to verify that no sensitive chunks containing either personal or confidential data, are fed into the generation model. Sensitivity checks can be performed with transformer-based classification models, which are trained to recognise sensitive data (Petrolini et al., 2022). These models compute a sensitivity score for each content chunk, and any chunk that exceeds a predefined threshold is flagged and excluded from the pipeline.

Summarisation
In certain cases, it can be beneficial to summarise the results from the top n chunks. It helps remove potential redundancy, since chunks may overlap. Overlapping chunks would consume additional tokens during generation, incurring unnecessary LLM costs. A disadvantage of summarisation is the loss of citation fidelity and traceability, which makes it harder to attribute specific statements to their source (Zhu et al., 2024).

Reordering
Another effective post-processing strategy is to reorder the top-n chunks before generation. LLMs often miss information located in the middle of the provided prompt. Therefore, chunks with the highest score should be placed at the beginning and end of the prompt, reducing the probability that the LLM will miss critical information (Liu et al., 2023).

Generate Answer
When the final top n chunks have been determined, the main LLM is called to generate the answer to the modified query. The prompt typically comprises multiple elements designed to maximise answer quality and reliability. Frequently passed information includes:

- Instructions on what the model must do.
- The original query plus the query modifications.
- The retrieved top-n chunks.
- Some guardrail instructions to prevent the model from giving unwanted answers.
- Details about the format in which the answer should be returned, for example, in JSON.

This structure ensures that the LLM has sufficient context to understand the task it has been given.

Post-process Answer
After generation, it is often beneficial to perform final refinements (Akkiraju et al., 2024; Packowski et al., 2024). During this step, the answer can be fact-checked to identify and correct hallucinations. The chance of hallucination increases if retrieved chunks contradict each other or if they contradict the internal pre-trained knowledge of the LLM itself. Citations are often added to allow humans to fact-check

generated content themselves, enhancing system trust and accountability. The answer format can also be refined by correcting minor JSON or HTML errors introduced during generation.

Interaction Model
Organisations also need to decide how users will interact with the RAG system. A search-style interaction supports single-query answering, meaning there is no memory of previously asked questions. This method thus requires no session management and simplifies overall implementation. Search-style interactions support level-1 and level-2 queries, but they limit the user experience.

A more complex approach is the use of a conversational interface that allows multi-turn dialogue. Users can ask follow-up questions and reason together with the system. They are already familiar with this concept, as commonly used commercial LLMs such as ChatGPT and Gemini also utilise this interaction method. It enables answering queries at all levels and provides a more natural way to converse with the system. However, multi-turn dialogue increases complexity and overall costs, as the conversation history must be stored and passed to the model as additional context, leading to higher token consumption.

Architecture of Agentic RAG

The last paradigm focuses on AI agents and is called Agentic RAG (Singh et al., 2025; Y. Gao et al., 2025). Agents are instances of an LLM that orchestrate multiple tasks. The agent can reason about the tasks required to reach the desired goal and autonomously execute them. This contrasts with Advanced RAG, in which the system follows a predetermined path. Instead of taking the query and retrieving its most relevant data, the agent first devises an action plan and sequentially executes the identified steps. If during execution the desired results are not found, the agent can deviate from its original plan and create a new one. This reasoning capability enables the answering of more complex questions that require reasoning over multiple sources and utilise external rationale. Agentic RAG is goal-oriented and can quickly adapt to changes. It attempts to dynamically determine the most effective way not only to retrieve relevant information but also to actively achieve the stated goal. When comparing Agentic RAG to the query complexity levels, it should be able to handle all four levels if implemented well.

Agentic RAG often makes use of Chain-of-Thought (CoT) reasoning to guide the orchestration process. CoT reasoning entails splitting the original query into multiple sub-queries, which together can deliver an answer (Wei et al., 2023). Each sub-query is executed sequentially, with its retrieval results influencing subsequent retrievals. By explicitly modelling intermediate reasoning steps, CoT provides RAG with a structured approach to handle complex queries that require reasoning across separate data sources.

Note that Agentic RAG utilises the same components as Advanced RAG, except that an LLM now controls the retrieval strategy, dynamically reacting to intermediate results in multi-hop retrieval. The orchestration agent can recognise the complexity level of a query and act accordingly. We show a simplified single-agent RAG architecture in Figure 2.6. Note that many steps that were introduced in Figure 2.5 are excluded here to avoid cluttering the diagram. In single-agent RAG, the orchestration agent is typically a small, fast LLM. For the final answer generation, a larger LLM is required to synthesise the retrieved context.

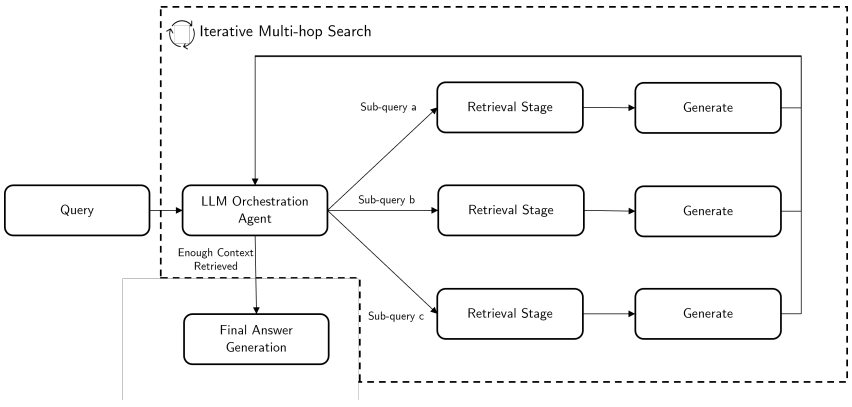

Figure 2.6: A simplified single-agent RAG architecture, with an LLM as the orchestration agent.

A different approach, which leverages several agents, is called Hierarchical Multi-agent RAG (Singh et al., 2025). In this method, each retrieval step is handled by a specialised agent, e.g., one agent for SQL tables, one for vector database search, and one for the internet. Each sub-agent orchestrates its own retrieval process, analogous to a single agent using CoT, as shown in Figure 2.6. These agents perform their tasks in parallel, meaning they are independent of one another. The sub-agents report to the master agent, who oversees the query-answering process. It determines which agents to invoke and when, and combines their responses to produce the final results. The master agent is often a large LLM, while its sub-agents are small and quick. This multi-agent approach enables the handling of complex queries, such as those at levels 3 and 4.

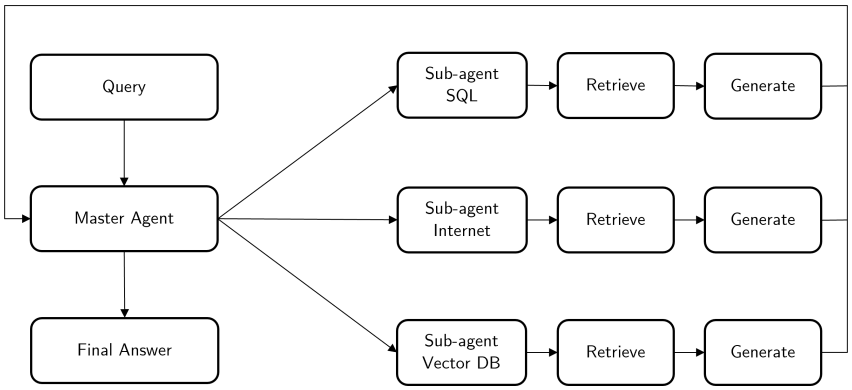

Figure 2.7: A simplified Hierarchical Multi-agent architecture.

For this thesis, we scoped Agentic RAG up to hierarchical multi-agent architectures. We considered the two discussed variants sufficient to illustrate the key principles and capabilities of Agentic RAG. Other variants were excluded because they did not provide further insight for our research.

RAG Paradigms and KBDC

So far, we have discussed KBDC capabilities, query levels, and RAG paradigms. We can now synthesise these concepts to show how technical paradigm choices enable or constrain strategic capabilities. The chosen RAG paradigm determines which query levels can be answered, which in turn determines what KBDC capabilities the system can potentially support. Table 2.4 presents these relationships, which are based on the combination of previously discussed sections.

Table 2.4: Technical RAG paradigms and their enablement of KBDC capabilities.

Paradigm	**Query Level**	**KBDC Capabilities**	**Primary Constraints**
Naive RAG	Level 1	Integrating (limited)	Single-hop retrieval, limited reasoning, basic semantic search
Advanced RAG	Levels 1–2, 3 (limited)	Integrating, Replicating, Assimilating*, Imitating*	Predetermined paths, limited multi-hop reasoning
Agentic (Single)	Levels 1–3	Integrating, Replicating, Building (limited), Assimilating*, Imitating*	Single-agent synthesis limits, governance challenges
Agentic (Multi)	Levels 1–4	Integrating, Replicating, Building, Assimilating*, Imitating*	Architectural complexity, governance challenges

** Requires internet search or external API access.*

The relationship between paradigms and their associated capabilities is hierarchical. More complex paradigms retain the functionalities of their predecessors while adding extra features. Naive RAG only supports limited integrating, but is cheap, easy to build, and simple to maintain. Advanced RAG has the potential to support multiple KBDC capabilities but requires more sophisticated development expertise and introduces retrieval complexity that organisations with limited technical capacity may struggle

to manage. However, Advanced RAG represents a broad spectrum of methods, making it difficult to quantify in terms of enablement. Agentic RAG introduces the dynamic capabilities that are necessary to support building, but demands substantial IT and AI expertise (see Section 2.3.4).

However, these capabilities represent technical potential rather than guaranteed outcomes. Organisations cannot incrementally upgrade from simpler to more complex paradigms without substantial changes in system architecture. Moreover, industry analysis revealed that advanced RAG techniques often face infrastructure immaturity and scalability constraints that limit their enterprise viability (RAGFlow, 2025), making simpler RAGs strategically preferable when organisational capabilities cannot support complex implementations. The query complexity required to achieve desired KBDC capabilities thus represents a key consideration in RAG implementation

2.3.2 The Knowledge Base Constraints

The scope of the RAG knowledge base defines what information the system can access and directly shapes its answer coverage. Setting up a knowledge base requires several foundational choices, such as the content of the knowledge base and whether to grant the system internet access.

Knowledge Base Management

An important aspect is the management of the knowledge base's comprehensiveness. Organisations must decide which files to add to the knowledge base. An intuitive assumption might be that including more documents leads to better results, as coverage increases. However, prior studies have noted that while increasing the number of documents may enhance coverage, this expansion introduces several trade-offs (Bruckhaus, 2024; Barnett et al., 2024; Shen et al., 2024; Oche et al., 2025).

First, expanding the knowledge base increases the retrieval complexity. Storing more documents increases the number of stored chunks, increasing the likelihood of retrieving irrelevant chunks. Chunks may lie close together in the vector space while containing content about two similar yet distinct subjects. If both are retrieved, this could confuse the model, resulting in a faulty final answer that combines the two chunks. This challenge grows as the knowledge base expands, requiring more effective retrieval and re-ranking methods, which introduces additional computational overhead and implementation complexity.

Second, maintaining a knowledge base requires continuous effort (Oche et al., 2025). Newly added documents require verification for accuracy, relevance, and compliance. But organisations change as processes, people, and policies evolve over time. Files that were accurate at the time of ingestion may become outdated and need to be replaced or supplemented with the most current information. The quality of RAG output is, just like regular machine learning models, constrained by its input data; inaccurate, duplicate, or outdated data will yield unreliable results, regardless of the RAG pipeline's sophistication (Hasan et al., 2025; Ouyang et al., 2025; Müller et al., 2025).

Solutions such as real-time updates ingest documents as they are created or modified, ensuring total freshness. However, this approach requires an ingestion pipeline directly connected to all source systems, which increases operational complexity. Another solution is to ingest changes in batches, such as daily or weekly, to minimise performance impact. It does introduce some information staleness, and whether this is appropriate depends on the use case. A different approach is to perform manual refreshing, in which humans carefully curate what and when new files are ingested. This gives complete control over the content knowledge base, which is beneficial in use cases where strict quality measures apply. Curating the knowledge base is a continuous governance effort, and its overhead increases as the knowledge base expands.

Last, the scale of the knowledge base affects infrastructure costs (Ren et al., 2025). Storage and query costs increase with the number of chunks in the vector database. This is especially important when using third-party vector database vendors, which charge based on volume, data size, and service-level agreements, such as guaranteed uptime or query latency. While open-source alternatives may reduce licensing costs, they still require technical expertise and infrastructure costs that scale with the size of the knowledge base.

These trade-offs necessitate organisations to make strategic choices about their knowledge base that align with the intended system capabilities. A system designed for broad organisational QA may prioritise breadth over depth, only including frequently accessed files across multiple internal domains. A more domain-specific RAG solution may require comprehensive QA coverage, including all newly added domain-relevant files in its knowledge base.

External data access

RAG systems may extend beyond internal knowledge through integration with internet search. External data access enables integrating internal documents with academic research, regulatory documentation, and industry information. Internet access thus enables the potential use of external KBDC capabilities, such as assimilating and imitating. However, this addition introduces some downsides as well.

First, overall retrieval complexity increases, as the system must now route queries to the internet, the internal knowledge base(s), or both (Guerraoui et al., 2025). This requires additional mechanisms to handle the retrieval steps, especially in advanced RAG setups where the path is predetermined.

Second, it becomes more challenging for the model to determine the weights it should assign to different information sources (Jin et al., 2024). Internal and external sources may conflict, for example, due to data staleness in the knowledge base. It should also be clear which information in the final answer is internal and which is external, as this helps people assess the answer's correctness.

Third, the inclusion of internet access introduces additional security challenges (Ammann et al., 2025). For example, sensitive information retrieved during the API call could be exposed to the browser. In addition, queries might retrieve malicious code during internet scraping that may inject the knowledge base, see also Section 2.3.4.

Organisations that strive to improve only internal KBDC capabilities do not require internet access in their RAG system, although it may enhance them. In contrast, organisations pursuing external assimilating and imitating capabilities require the internet to access knowledge not yet held internally.

2.3.3 The RAG Infrastructure Constraints

Infrastructure choices determine what the RAG system can do, thereby affecting feasibility, performance, and costs. Infrastructure choices constrain RAG capabilities and are inherently difficult to reverse, as they underpin the RAG pipeline. Examples of decisions include retrieval architecture, deployment models, and technology stack selection.

Retrieval Method

Thus far, we have stated that RAG utilises vector databases for data storage, which employ numerical representations of text that enable similarity calculations. While this is the most dominant method for RAG, an alternative approach using knowledge graphs has been proposed, called GraphRAG (Edge et al., 2025). This method represents knowledge as nodes and edges that correspond to entities and their relationships. It extracts entities such as names, products, concepts, places, and events from chunks, which serve as nodes in the knowledge graph. These nodes are related to one another. For example, from text mentioning "OpenAI released ChatGPT," GraphRAG would extract: [OpenAI] –created–> [ChatGPT], forming two entity nodes connected by a "created" relationship edge.

Unlike vector-based retrieval, which treats chunks as independent items, GraphRAG constructs a single, large knowledge graph that includes all entities and relationships from ingested chunks. This unified knowledge graph enables reasoning about entity relationships across the entire knowledge base, thereby enabling more precise retrieval. Knowledge graphs are well-suited to multi-hop retrieval, particularly when uncovering relationships is critical. GraphRAG is therefore well-suited to handling more complex, relational queries (levels 3 and 4) that require reasoning across multiple sources.

However, GraphRAG introduces additional complexity compared to the vector databases (Peng et al., 2024; Han et al., 2025; Cheng et al., 2025). It requires accurate extraction of entities and relations from ingested chunks, where mistakes can snowball and affect the entire knowledge graph. To ensure correct

extraction, organisations are required to define entity types, relationship schemas, and extraction rules before constructing the graph, which requires advanced knowledge of both graph and domain-related concepts. GraphRAG also does not scale as easily as vector databases. As the knowledge graph grows, queries become more computationally expensive, leading to a slower and more costly RAG system. This limits its applicability for large knowledge bases. For organisations primarily pursuing building capabilities that require synthesis across documented relationships, GraphRAG may be worth the additional complexity.

However, most RAG applications utilise vector databases, as they are easier to implement, more scalable, and overall less expensive. Knowledge graphs are particularly well-suited for domain-specific RAG applications where the relationships between entities are crucial to satisfying the business case. We acknowledge that knowledge graphs are a critical RAG category; however, for simplicity, we will use vector databases as the primary retrieval architecture for the remainder of this thesis.

Deployment Models

Organisations also face a decision when choosing between on-premises, cloud-based or hybrid deployment models (Kotha, 2022). The choice determines how well the system can accommodate growth in knowledge base size and query volume. Cloud-based RAG offers rapid deployment, elastic scaling and reduced overhead costs for services such as vector databases, embedding models, and LLMs. These services introduce usage-based costs, create vendor dependencies, and raise questions about handling sensitive data in external environments. Vendor lock-in makes it more costly to switch between providers or adopt new technologies, thereby constraining future architectural flexibility (Opara-Martins et al., 2016). This dependence leaves organisations exposed to vendor roadmaps and pricing models, reducing their autonomy over features and costs.

On-premises deployments provide control over data handling, infrastructure, and cost management. It is especially relevant for domains with strict data regulations. Empirical research into deployments in healthcare and governance contexts has demonstrated that self-hosted infrastructure reduces GDPR risks and improves overall system speed (Hasan et al., 2025). Drawbacks of on-premises deployment include the need for infrastructure expertise, upfront investment, and limited elastic scaling. So while on-premises offers full infrastructure control, it comes with substantial overhead. Organisations with stable, predictable demand and available capital may find on-premises deployment more economical over time, particularly when data control is favoured.

Hybrid deployment attempts to combine the two models by distributing components between them. For example, data storage could be performed on-premises, while LLM inference is performed in a cloud environment. Hybrid deployment requires careful orchestration across different environments to ensure data protection and optimal system performance (Pan and Wang, 2025; Yuan et al., 2025).

Choosing the deployment model also depends on the expected token consumption and the expected life cycle. Research has found that local LLM deployment break-even occurs for small enterprises within three months of deployment, medium enterprises between three and 34 months, and large enterprises between four and 69 months[3](Pan and Wang, 2025). While the economic advantage of running LLMs locally is lower for large enterprises, the added security control may outweigh the extra incurred costs. However, models, cloud costs, and API costs are constantly changing, making LLM deployment not a one-off decision but a continuous process of adapting the architecture to the latest AI developments.

Technology Stack Choices

RAG components, such as vector databases, embedding models, and LLMs, can be procured from open-source repositories or third-party vendors. Proprietary services and commercial LLMs provide ready-to-use features with guaranteed performance and operational support, charging on a usage basis. These services offer ease of use and speed to market, but limit pipeline customisation and introduce vendor lock-in (Mohammadabadi et al., 2025). Organisations can also procure an off-the-shelf RAG solution in full, eliminating the need for development but severely constraining customisation.

[3]Break-even timing depends on model size, usage patterns, and the relationship between upfront hardware costs and API expenses. Calculations assumed that different-sized organisations use different model sizes. LLM prices and sizes are subject to change.

Open-source alternatives and self-hosted LLMs offer more control and negate the need to pay per usage. However, they introduce additional costs, including hardware, operational, and optimisation costs. Building an operational, scalable RAG solution is challenging, and organisations would be wise to assess their own capabilities before beginning development. This decision involves weighing the total cost of ownership against the convenience and predictability of proprietary services. Organisations that prefer minimal operational complexity and rapid deployment are more likely to engage third-party vendors, whereas organisations that prefer full control and have sufficient IT capabilities may opt for open-source solutions (Katsamakas and Xin, 2019; Oliveira et al., 2019).

Tools such as Microsoft 365 Copilot[4] and Google's[5] Gemini represent fully integrated AI tools within their respective ecosystems. They offer a comprehensive package that includes integrated infrastructure, data handling, and updates. While these systems do offer RAG functionality, it is not their core value driver. As fully managed services, they offer less architectural flexibility than custom-built solutions. These products will be beneficial for organisations that already fully utilise one of these ecosystems and whose RAG requirements align with the limited capabilities. Organisations that require custom capabilities, such as rigid data control or integration with legacy data sources, may opt to build their own solutions or procure dedicated RAG systems.

Infrastructure decisions are foundational choices that determine RAG capabilities, costs, and constraints during its lifecycle. Third-party software packages, such as Copilot, can enable rapid RAG deployment capabilities but constrain system flexibility due to a fixed RAG architecture. Organisations can build their own RAG systems from the ground up using commercial or open-source software, but this comes at the cost of increased complexity and overhead. The technology stack choice thus represents a strategic trade-off between deployment speed, costs, and system capability ceiling.

2.3.4 The RAG Governance Constraints

RAG governance ensures that RAG operates securely, ethically, and in compliance with relevant regulations. The RAG architecture determines what the system can do; governance states what it should do and who is accountable for its behaviour. It determines who has access to specific organisational resources, ensures compliance with data protection regulations, and makes RAG more accountable through the use of audit trails. Governance decisions are foundational because they shape the system's behaviour and are difficult to change once the system is operational.

Access Control

Access control is a critical challenge for knowledge bases. Information is often intended for separate organisational levels, with documents classified into confidentiality categories which are mapped to specific organisational roles or attributes. RAG inherits these access-control requirements when ingesting confidential files. Without safety measures, employees may access content they are not authorised to view, thereby violating access control policies. Failures could lead to unauthorised data exposure, regulatory violations or a decrease in system trust. Therefore, robust access control becomes necessary to ensure compliant RAG retrieval and generation (Bruckhaus, 2024; Oche et al., 2025; Ammann et al., 2025).

In practice, RAG systems often enforce access control through pre-retrieval filtering, ensuring that sensitive or unauthorised documents are excluded before generation. It constrains which chunks can be retrieved based on the users' permission levels. The system must be able to recognise the current user's role or attribute, leveraging existing identity management systems. These roles or attributes are then compared against metadata tags added during document ingestion.

The complexity of RAG access control expands as organisations grow. Small organisations with simple permission structures incur little additional overhead. Large organisations, which often have cross-functional projects, international branches, and complex confidentiality policies, face challenges in metadata enrichment. For example, adding metadata only at the document level might be simpler, but it lacks the flexibility of chunk-level tagging. While access control metadata can often be inherited from

[4]https://copilot.microsoft.com/
[5]https://gemini.google.com/

the original data sources, this will not always be the case. Manually adding missing metadata is not a scalable solution, but organisations can prepare alternative enrichment methods, such as LLM-based or traditional ML classification (Bouabdallah et al., 2021).

Organisations also experience continuous changes in access control as employee roles change and documents are reclassified. Real-time synchronisation with source systems maintains consistency, but introduces additional infrastructure dependencies (Mickie and Elson, 2025). Synchronising access control in batches reduces complexity, but leads to permission staleness, opening up the risk of unwanted data exposure.

Data Privacy and Regulations

While access control determines who can view certain information, data regulations define what information may be used. RAG solutions that process personal data must comply with regulations such as the GDPR and HIPAA (Bruckhaus, 2024). Personal data refers to any information that relates to an identified or identifiable individual, and includes names, identification numbers, and location data (European Parliament and of the Council, 2016). While the full exclusion of personal data in a knowledge base simplifies overall compliance, some domains, such as HR or healthcare, require its use to be functional. Personal information is also typically found in documents such as presentations, notes, and reports. Excluding these files reduces system usability, and removing personal data manually is not scalable. Organisations face a foundational strategic choice: adopt risk-averse approaches that exclude personal data and consequently limit use cases, or pursue approaches that include personal data but require complex compliance mechanisms. This decision shapes the scope of the knowledge base, infrastructure options, and governance overhead.

The risk-averse approach entails the full exclusion of personal data from the knowledge base. Before ingestion, files require pre-processing by masking all instances of personal data. This can be achieved by machine-learning classifiers trained to recognise personal data (Petrolini et al., 2022). Upon detection, personal data can be excluded by replacing detected content with generic tags.[6], excluding paragraphs or full document rejection. Each strategy captures different levels of context, with tagging being the most context-preserving method. However, classifiers are not perfect, and some personal data may still be included in the RAG knowledge base due to misclassification. This means that even in risk-averse implementations, continuous monitoring is still necessary.

If RAG use cases require personal information, such as in HR or healthcare, regulatory compliance needs to be built in from the start. Compliance often requires architectural mechanisms for data deletion, infrastructure constraints from data residency rules, and enhanced access controls (Tikkinen-Piri et al., 2018). Including personal data in the knowledge base creates more opportunities to generate value, but introduces additional governance overhead.

Audit Trails

Organisations may comply with data protection regulations, but are still required to demonstrate compliance. This necessitates mechanisms that make the system transparent and verifiable, which is typically achieved through audit trails (Fernsel et al., 2024; Li et al., 2025). Audit trails record system activity, enabling accountability, monitoring, compliance verification and more comprehensive testing. Designing trails involves what the system tracks (e.g., timestamps, user IDs, retrieved chunks) and how long it retains saved logs.

Extensive logging serves two purposes. For compliance and governance, it provides accountability and enables audit verification of access control, decisions and system behaviour (Xu et al., 2025). For system optimisation, comprehensive logs enable precise debugging, parameter testing, and the evaluation of pipeline changes by capturing full system behaviour. Logging also has its trade-offs. Comprehensive logging requires additional storage and governance overhead, especially if retrieved chunks and generated answers are logged. If an organisation opted to use personal data in its RAG knowledge base, the logged chunks and answers would require the same extensive governance systems, resulting in additional overhead.

[6]Detected personal names or identifiers are anonymised. For example, “Menno Bruinsma” is replaced with [NAME]

Building audit trails is a critical part of enterprise RAG, since historical system behaviour cannot be captured in retrospect. If it is concluded that specific audit trails are required for compliance while the system is already operational, it is impossible to reconstruct previous operational behaviour.

RAG Security

While audit trails enable retrospective accountability, they do not prevent malicious interference. Securing the RAG pipeline against adversaries seeking to influence RAG behaviour in unwanted ways is an essential element of the RAG lifecycle (Ammann et al., 2025).

RAG knowledge bases contain valuable organisational knowledge, enabling rapid retrieval and reasoning about projects, products, and processes. This makes RAG an attractive target for adversarial actors. For example, hostile external or internal actors might attempt to extract confidential data from the RAG system by circumventing implemented access controls. Empirical research into RAG systems and practitioners found that developers consider security the most important element of RAG (Brehme et al., 2025).

RAG systems have several categories of security risks (Ammann et al., 2025):

- Retrieval-related risks: Sensitive information might be unintentionally exposed through data retrieval or embedding processes.
- Operational risks: Disclosure of confidential data during prompting or interactions with cloud-hosted LLMs.
- Data-manipulation risks: Malicious prompt injections or tampering with the knowledge base to alter model outputs.

These risks reflect the need for a multi-layered defence strategy that combines secure data storage, controlled retrieval, and prompt guardrailing (Ammann et al., 2025). Techniques such as query verification, encryption, access control, and post-retrieval filtering can mitigate attack risks.

The different RAG paradigms present distinct security profiles. Advanced RAG, with its predetermined execution paths, enables consistent control and verification of system behaviour. In contrast, Agentic RAG employs autonomous agents that orchestrate dynamic execution paths, introducing additional security complexity (Lupinacci et al., 2025). This architectural difference affects the feasibility of implementing robust access control and monitoring mechanisms. Similarly, external internet access capabilities expand system functionality while introducing additional security considerations regarding data exposure and malicious content injection (Tan et al., 2024).

Security implementations involve trade-offs between protection and performance (Zhang et al., 2025). Additional safety measures enhance system robustness while introducing latency and computational overhead. Similarly, strict access controls reduce data leakage risks but may limit KBDC capabilities by restricting document accessibility. Comprehensive logging enables audit capabilities at the cost of storage and governance complexity.

Together, access control, data privacy, audit trails, and security form the four core pillars of RAG governance. Each pillar governs a distinct risk dimension, yet they operate together to shape RAG system boundaries. Governance serves as both an enabler and a constraint on KBDC capabilities. Strict access controls and compliance mechanisms enable RAG to scale across organisational boundaries while maintaining security, but also introduce latency and complexity that may limit operational usage. Governance overhead and KBDC capabilities present an inherent trade-off, with organisations balancing security requirements against system performance and functionality.

Chapter 3

Method

This chapter describes the methods we used to address our research question and its three guiding questions, as stated in Section 1.2. We explain our inductive reflexive thematic analysis approach and describe how we progressed through its various stages. We conclude by discussing the four quality criteria for qualitative research, researcher positionality, and ethical considerations.

3.1 Research Approach

The main methodology we used to answer our research question is inductive reflexive thematic analysis (Braun and Clarke, 2006). This qualitative method involves systematically coding interviews to extract themes that capture overall patterns in the collected data. In this context, inductive means that codes were directly derived from the interviews, rather than being predetermined by a fixed codebook. This way, our codes were inherently flexible and could be modified as needed. The reflexive aspect refers to the process of coding interviews throughout the entire data collection period, rather than coding after all interviews have been completed. Accordingly, we started coding as soon as the first interview was completed. This reflexive approach enabled a deeper understanding of the data, allowing us to ask more informed questions during interviews. Had we waited to code until all interviews were completed, we would have risked losing contextual details that arose during them. It allowed us to remain flexible and respond quickly when something interesting but unexpected arose during interviews. Figure 3.1 shows the different phases we went through during our thematic analysis.

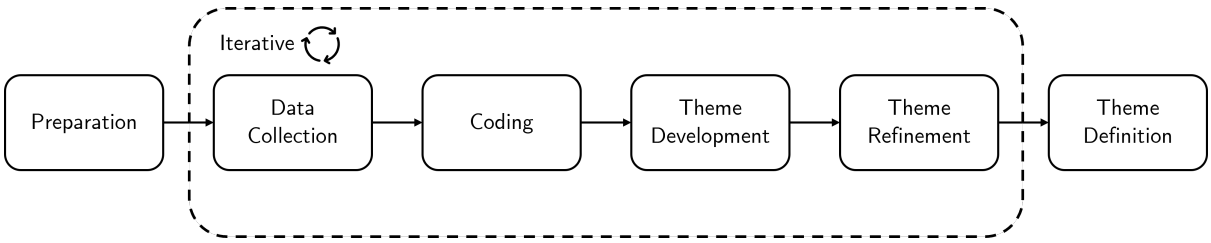

Figure 3.1: The standard research flow of thematic analysis.

3.1.1 Preparation

We began our research by building RAG prototypes from scratch on local devices using open-source LLMs. This hands-on experience, which took approximately one week, proved essential for several reasons. First, it enabled us to understand the technical trade-offs that practitioners must make in practice. Second, it provided context to better understand technical papers on RAG. Third, it established credibility during interviews, as we could engage fully with technical experts on challenges, solutions, and maturity views. Participants responded positively to our technical understanding, resulting in richer discussions.

Based on our literature review, we developed a semi-structured interview guide (see Appendix C) to extract information on RAG strategy, design decisions, trade-offs, and constraints. We refined the guide based on feedback from strategy consultants to ensure that questions effectively addressed our research objectives.

We chose semi-structured interviews because they offer a flexible approach to data collection. This approach allows interviewers to select and sequence questions based on emerging insights rather than following a rigid pre-determined script. While surveys could have provided quantifiable data, we determined they would not yield the contextual depth required to address our research question. Before beginning the actual data collection, we conducted a test interview with an AI consultant to validate our questions and technical understanding. The test interview's success indicated that we were ready to proceed to the data collection phase.

3.1.2 Data Collection

Sampling Strategy

As stated in our research scope in Section 1.3, we focus on large organisations that are active in the Netherlands. These organisations possess strong financial resources and development capacity, which are required to deploy large-scale RAG solutions (Yang et al., 2024). In addition, they often possess large, complex knowledge repositories, making RAG an attractive application for knowledge management.

We used purposive expert sampling to recruit participants with direct experience designing or implementing RAG systems in large organisations. This approach ensured engagement with individuals possessing either technical, organisational, or strategic expertise. In practice, most interviewees demonstrated familiarity across all three domains. We sampled variation in factors such as solution scope, organisational role, and industry sector. Given the exploratory nature of our research, which prioritised empirical understanding of RAG practices over establishing statistical generalisability, this sampling strategy proved appropriate.

We conducted expert 14 interviews over three months (see Table 3.1 for the full overview of the participants). Interview 11 involved two experts simultaneously but is treated as a single participant for analysis purposes. We recognise that a large number of interviews are sourced from Big Four consulting; however, most consultants worked on multiple projects across different sectors and organisational contexts. Combined, they have worked for over 15 different organisations. This provides a cross-organisational perspective, a view unavailable to regular practitioners. However, we acknowledge that this cross-organisational perspective may be biased towards a RAG consulting perspective. We decided to stop reaching out to potential participants after interview 14, as we determined that (i) we reached theoretical saturation, and (ii) confirmed the emerged patterns via interviews 9-14 (see Section 3.2).

Interview Method

Interviews were conducted in person and via Microsoft Teams, ranging from 30 to 75 minutes. All interview recordings were made via Microsoft Teams because it supports both Dutch and English transcriptions. If interviews were in person, we would place a laptop in the middle of the table and join a dummy Teams meeting to enable transcription. We often still needed to manually refine transcriptions to accurately represent the interview content, especially when the interview was conducted in Dutch. After these improvements, the interviews were ready for manual coding.

Table 3.1: Overview of interview participants (n=14)

ID	Role	Organisation	Sector	RAG Scope
P1	ML Engineer	High-tech Manufacturing	Semiconductor	Domain
P2	Innovation Manager	Big Four Consulting	Professional Services	General
P3	Product Owner	Research Institution	Higher Education	General
P4	AI Consultant	Big Four Consulting	Public and Cybersecurity	Domain
P5	AI Developer	Big Four Consulting	Professional Services	General
P6	ML Engineer	Research Infrastructure Provider	Higher Education	General
P7	AI Consultant	Big Four Consulting	Energy and Telecommunication	Both
P8	AI Consultant	Big Four Consulting	Retail and Logistics	Both
P9	Director of AI	Software Vendor	Enterprise Software	Both
P10	AI Consultant	Big Four Consulting	Tax and Legal	Domain
P11	Data Scientist and ML Engineer	Financial Institution	Financial Services	Domain
P12	AI Consultant	Big Four Consulting	Professional Services	Domain
P13	Data Scientist	Big Four Consulting	Tax and Legal	Domain
P14	AI Consultant	Big Four Consulting	Professional Services	Both

Domain refers to a single functional domain (e.g., legal, HR); General refers to multiple functional domains; Both indicates the participant worked on projects spanning both domain-specific and general RAG implementations.

Our reflexive approach enabled us to refine the interview design during data collection. Early interviews (P1-P4) were too focused on technical architecture, while it became clear that practitioners did not see this as the main barrier. This led to fewer questions about technical factors (e.g. operational design decisions), with more focus shifting to strategic alignment, organisational challenges, and future outlooks.

3.1.3 Coding

We performed the coding process in ATLAS.ti [1]. We always began coding an interview within 3 days of its occurrence to preserve contextual freshness. After coding the first four interviews, we recognised that the codes were overly focused on technical details and recoded all of them from scratch. Following interview seven, we recoded everything again, resulting in a completely new codebook that captured more details and topics. It also became clear that some concepts from earlier interviews were not particularly important, leading to the dropping or refactoring of codes. After interview 12, we again refined our codebook and recoded all interviews to more consistently capture insights. After the last interview, we performed a final pass to ensure that our codes accurately represented the interview data.

To manage the large number of codes, we organised them into code groups that correspond to practical topic areas such as strategy, governance, and adoption. These code groups were used to facilitate code review but were not used for theme development. We used a systematic naming convention for our codes: a prefix indicating the code group (e.g., DATAQUAL_ for data quality, STRAT_ for strategy, GOV_ for governance), followed by descriptors that, for example, distinguish among challenges, success factors, and solutions. For example, DATAQUAL_Success-factor_Standard-format indicates a data quality success factor regarding standardised data formatting. Code groups were identified naturally throughout the coding process rather than being predetermined by our literature review and the frameworks we used, though they align with findings from the literature in Chapter 2. Each code is also accompanied by a definition clarifying its scope and context, which evolved gradually throughout the recoding process.

[1] ATLAS.ti is a qualitative data analysis tool used for coding, memoing, and organising qualitative data. Available at https://atlasti.com/.

3.1.4 Theme Development, Refinement, and Definition

To develop themes, we examined how the various codes represented overarching concepts in the data. All individual codes were reviewed and mapped to one or more potential themes based on their contextual similarity. Themes are thus represented by a cluster of codes that address similar aspects of RAG implementations. As noted earlier, code groups were not used for theme development but rather as an organising tool.

Each main theme was then subdivided into subthemes to capture the distinct concepts within the overarching theme. For example, theme 1, Strategic Motivations for RAG Adoption, consists of three subthemes. 1.1: RAG as a Means for Improving Productivity, 1.2 RAG as a Means for Knowledge Discovery and 1.3 RAG as a Means for Creating Future AI Capabilities. This hierarchical structure allows analysis of the broad overarching theme and the various smaller concepts that lie within it.

Theme refinement was conducted iteratively through multiple rounds of code assignments. We validated the final themes by assessing whether each code contributed unique insights to the theme. Once we determined that the themes and subthemes captured the overarching concepts in our data, we could proceed to write our results.

3.1.5 Framework and Design Principle Development

Following theme identification, we extended our analysis to develop a framework and design principles. Given the exploratory nature of our research, these artefacts emerged naturally from our empirical findings rather than being designed through deliberate artefact creation methodologies such as design science (Hevner et al., 2004).

We developed an integrated framework that captures the interactions among the different themes; see Chapter 5. This led to the development of a framework comprising four layers, each grounded in empirical evidence. It synthesises our findings with the KBDC, TOE, and TAM frameworks to explain how the different layers interact dynamically within the constraints identified in our data.

We also developed 11 design principles that capture best practices and common pitfalls articulated by interview participants. These patterns were generalised into principles, which we verified by checking existing scientific and grey literature. Each principle was refined to ensure actionability and distinctness from other principles.

3.2 Research Quality

We ensured rigour in our research by following the four quality criteria for qualitative research, namely: credibility, dependability, confirmability, and transferability (Guba and Lincoln, 1985).

Credibility
We ensured credibility by conducting 14 expert interviews over three months (October-December 2025), who in total worked on 15+ RAG systems. Our reflexive coding approach enabled us to iteratively refine our coding to capture emerging interview results, ensuring that our research is grounded in the collected data. We continued data collection until saturation was reached, after which additional interviews would yield only limited new insights regarding strategy, design, and constraints. Saturation became evident around P9, where participants consistently confirm patterns from earlier interviews rather than introducing new ones. This saturation is shown in Figure 3.2, where after interview 8, only one new code emerged. Interviews 9-14 served as confirmation interviews, validating our findings and enriching existing codes with additional supporting evidence. Notably, some of these later interviews proved highly insightful, as our deeper understanding of enterprise RAG enabled us to ask more informed and targeted questions to confirm the patterns we found.

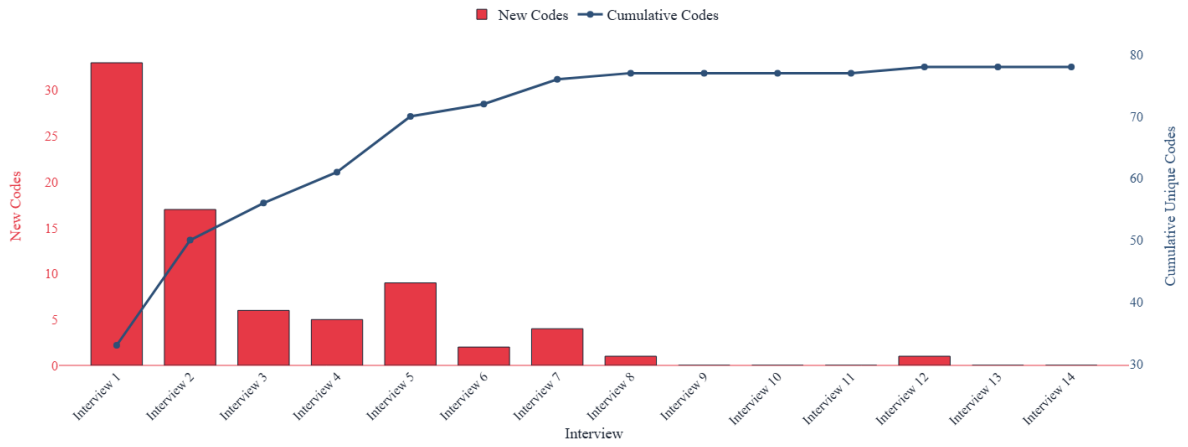

Figure 3.2: Code saturation over the 14 interviews. Saturation was reached at interview 9.

Dependability
Although our interview guide was semi-structured and could therefore vary across interviews, it enabled us to consistently cover the same core topics (strategy, design decisions, constraints). The multiple coding rounds ensured that the same analytical method was applied to all interviews. As a result, early interviews were coded with the same depth as later ones.

Confirmability
Interview transcripts, coded segments, and thematic development processes were preserved in ATLAS.ti and our custom data visualisation tool, creating an auditable trail from raw data to final themes. Themes arising from our thematic analysis are supported by multiple data extracts from various participants. Furthermore, we triangulated interview findings with academic and grey literature on RAG to verify whether they align with known challenges and solutions. We maintained reflexivity by continuously questioning our own assumptions, best demonstrated by our shift in focus from technical aspects to organisational considerations.

Transferability
Our sampling strategy yielded results from a diverse group of participants, each with different roles (developers, consultants, managers), sectors (consulting, finance, education), and implementation scopes (domain and general). We grounded the findings in existing frameworks such as KBDC and TOE, thereby enhancing the likelihood that they can be transferred across large organisations, regardless of location or sector.

3.2.1 Researcher Positionality

Our reflexive thematic analysis methodology requires researchers to be aware of their own potential biases. The researchers involved have backgrounds in computer science and machine learning. This, combined with the technology-oriented scientific literature, led to an early bias towards optimising RAG architectures. During early interviews, we quickly discovered that technical aspects were not the main determinants and that focus should shift more towards the layers around RAG. Recognising our technical bias enabled us to remain open to non-technical insights, ultimately strengthening our research.

Our positionality was further shaped by our practical RAG experiences within our research workflow. Beyond building basic RAG prototypes at the start of our research, we also utilised a commercial RAG platform, Claude Projects[2], to help organise and analyse our anonymised interviews. We chose a vendor tool over custom development due to immediate deployment, ease of use, and powerful LLM reasoning capabilities. We thus faced similar decisions as described in Chapter 2 and those articulated by practitioners in Chapter 4. While we recognise that our limited use case differs fundamentally from enterprise contexts, this decision-making process further increased our understanding of the trade-offs practitioners

[2]Claude Projects is part of Claude, a leading LLM AI assistant created by Anthropic. Available at: https://claude.ai

navigate.

Throughout this research, we employed LLMs as assistants for literature search, programming, conceptual understanding, and intellectual sparring. However, all research activities, such as literature synthesis, interview design, coding, thematic analysis, framework development, and interpretation, were conducted by the researchers. While LLMs provided valuable support, they never replaced human judgment throughout the research process.

3.2.2 Ethical Considerations

We followed standard guidelines for research involving human participants. Participants were informed in advance of the interview's goal and purpose, and were notified that the interview would be recorded for analytical purposes. During interviews, we again confirmed that it was acceptable to record the meeting before transcribing. All participating people and organisations were anonymised in the final results, and the data was used solely for academic purposes. Transcriptions were not shared with the supervising company.

Chapter 4

Results

This chapter presents the results from the thematic analysis of the 14 semi-structured interviews described in Chapter 3. Our analysis yielded 78 codes, each capturing distinct aspects of RAG implementations in enterprise settings. To better manage these codes, we organised them into 11 code groups that represent practical topic areas such as strategy, governance, and adoption. These code groups were not used as analytical units or themes themselves, but merely acted as organisational tools. The resulting organisational structure is shown in Table 4.1. Code counts are reported for transparency and do not imply analytical importance, in line with the principles of reflexive thematic analysis (Braun and Clarke, 2006). The complete codebook is shown in Appendix E, Table E.1.

Table 4.1: Organisational structure of codes.

Code Group	Topic	Number of Codes	Total Quotations
ADOPT	User Adoption	11	105
BUILD	Build Versus Buy	7	46
DATAQUAL	Data Quality	6	53
DEVELOP	Development Process	9	103
EVAL	Evaluation	6	33
FUTURE	Future Outlook	4	39
GOV	Governance	7	81
KB	Knowledge Base	7	54
KBDC	Knowledge-Based Dynamic Capabilities	5	36
STRAT	Strategy	11	113
TRADEOFF	Trade-offs	5	68
Total		**78**	**731**

From Codes to Themes

While the 11 code groups provided organisational structure, our analytical focus was on identifying RAG implementation patterns across our interview data by examining individual codes. Following Braun and Clarke's (2006) reflexive thematic analysis method, we iteratively constructed multiple themes from identified patterns across our data.

Through this analysis, we identified three distinct patterns during initial candidate theme development: codes relating to organisational motivations and strategic goals (primarily from STRAT, KBDC, FUTURE groups), codes describing implementation choices and trade-offs practitioners faced (primarily from BUILD, KB, DEVELOP, TRADEOFF groups), and codes capturing limiting factors that shaped these choices (primarily from GOV, DATAQUAL, EVAL, ADOPT groups). This led to the creation of three candidate themes, which revolved around (i) why organisations adopt RAG, (ii) what type of decisions practitioners face, and (iii) which contextual factors constrain enterprise RAG.

We refined these candidate themes in the second round by assessing coherence and conceptual exclusivity, leading to revisions to both individual code assignments and theme definitions. We conducted a

final review by examining both code assignment and thematic structure to assess whether they captured critical overarching patterns. This process led to the creation of three main themes, for which the code heatmaps are provided in Appendix D. We identified:

- *Theme 1: Strategic Motivations for RAG Adoption*
  Examines why organisations adopt RAG, revealing three primary strategic objectives: productivity improvement, knowledge discovery, and building future internal AI capabilities. Constructed primarily from codes in STRAT, KBDC, and FUTURE code groups.
- *Theme 2: RAG Design Decisions*
  Identifies design decisions that practitioners face when implementing RAG. Each decision involves trade-offs and has no universally optimal solution. Constructed primarily from codes in BUILD, KB, DEVELOP, and TRADEOFF code groups.
- *Theme 3: Constraints in RAG Implementations*
  Examines technical, organisational, and environmental factors that constrain RAG implementations. Organisational factors emerged as more binding than environmental and technical ones. Constructed primarily from codes in GOV, DATAQUAL, EVAL, and ADOPT code groups.

4.1 Strategic Motivations for RAG Adoption

Organisations primarily adopt RAG because of three strategic reasons: (i) improving productivity, (ii) improving knowledge discovery, and (iii) the desire to create internal AI capabilities. Figure 4.1 shows the code network for this theme, communicating which codes are related to it. The individual code labels have been removed for visual clarity. A heatmap illustrating the distribution of codes across interviews for this theme is provided in Appendix D, Figure D.1.

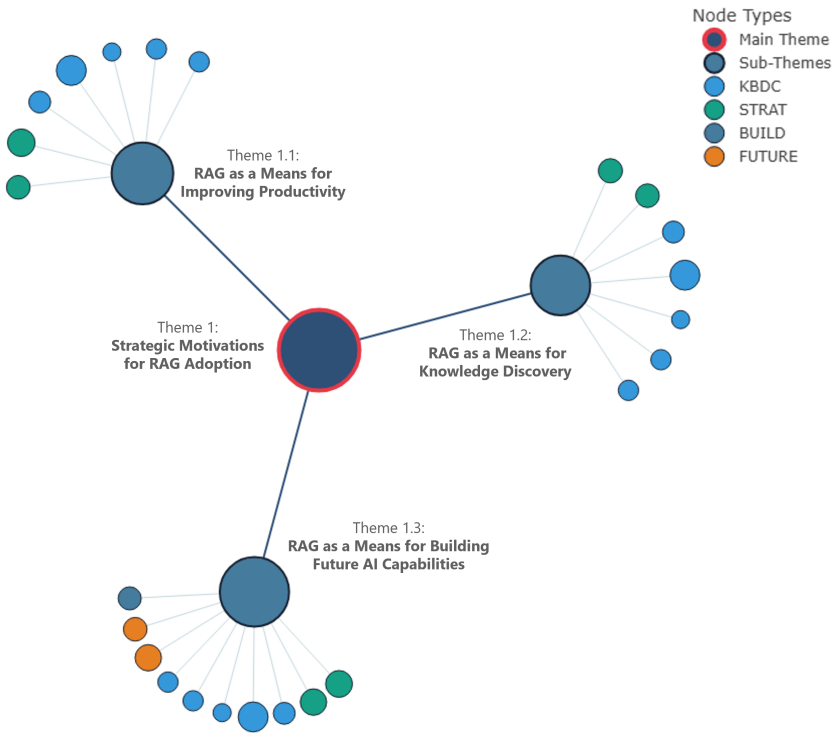

Figure 4.1: The code network for the theme: Strategic Motivations for RAG Adoption. Node size reflects relative code frequency.

4.1.1 RAG as a Means for Improving Productivity

Productivity improvement through reduced search time emerged as a dominant strategic motivation for RAG adoption, mentioned in 9 interviews. Participants described time spent searching for information across multiple disconnected systems as a clear inefficiency. RAG addresses this by enabling natural language queries, eliminating the need to know where information resides or how systems are structured. Important codes for this theme are: *STRAT_Motivation_Productivity*, *STRAT_Motivation_Value_From_Data*, *KBDC_Integrating*, and *KBDC_Replicating*. The code frequencies for theme 1.1 are shown in Figure 4.2.

Participant P1, an ML engineer at a semiconductor company, illustrated how RAG reduces this inefficiency:

> *"The application is used to enable users to query internal knowledge sources like their meeting transcriptions, their requirement documents, code bases, and their email. They can ask questions with natural language, which makes the process more efficient."*

The productivity improvement is clearly reflected by question-answering scenarios. Participant P8, an AI consultant working in retail and logistics, provided a concrete example:

> *"RAG is useful for resolving as many employee questions as possible, especially those they cannot find answers to in the portal. For example, if an employee has a question about their bicycle scheme, the bot can easily answer it by retrieving the correct document. That way, they get an answer immediately, instead of spending half an hour searching through a portal. That always delivers value."*

Beyond simple time-saving, organisations also saw the advantage of freeing employees from tedious work, allowing them to reallocate time to higher-impact, more fulfilling activities. Participant P10, an AI consultant specialising in the legal domain, articulated this dual objective:

> *"It starts with a clear vision of what we want to achieve with AI. It is an efficiency improvement. But we also simply want the work of our people to become more enjoyable. We want our employees to be able to focus more on the human contact or more high-end work."*

In addition to supporting individual human workers, RAG also augments AI agents by providing grounded knowledge for automated processes, further improving productivity within the organisation. Participant P12, an AI consultant at a Big Four company, explained how these purposes converge:

> *"RAG is a tool to improve agent accuracy on the one side, which is useful for process automation. On the other side, it helps people find knowledge. Searching for knowledge is always a problem within companies; it always takes a lot of time, including for our company."*

These findings reflect an organisational preference for integrating and replicating capabilities, enabling internal knowledge to be identified, transferred, and applied more efficiently.

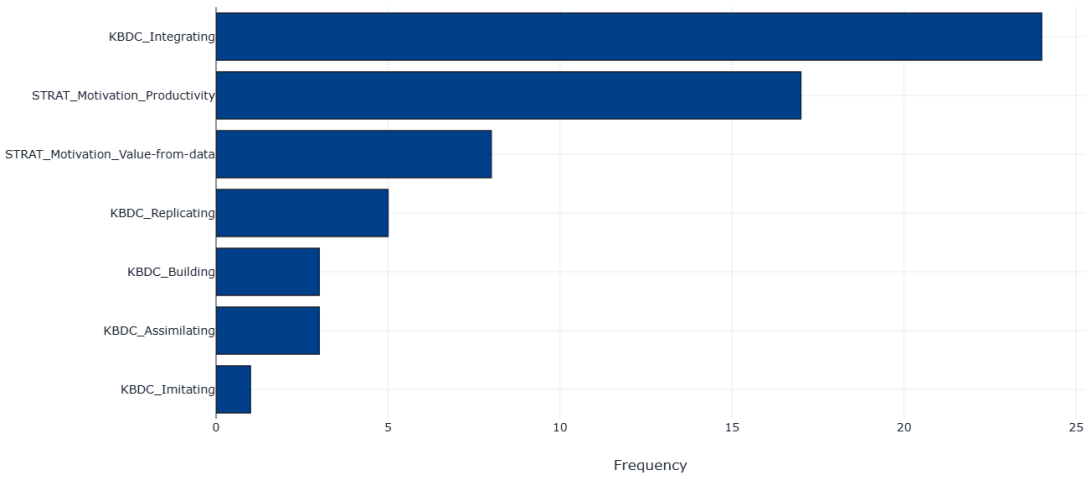

Figure 4.2: Frequency of codes within theme 1.1: RAG as a Means for Improving Productivity.

4.1.2 RAG as a Means for Knowledge Discovery

All participants explicitly or implicitly mentioned the reduction of information silos as a strategic motivation for RAG adoption. Information silos emerge when knowledge is fragmented across departments, geographic locations, document repositories, and various systems. RAG allows employees to find knowledge more easily across these organisational boundaries. Key supporting codes for this theme are: *STRAT_Motivation_Breaking_Silos*, *STRAT_Motivation_Value_From_Data*, *KBDC_Integrating*, and *KBDC_Replicating*. The code frequencies for theme 1.2 are shown in Figure 4.3.

Participant P2, an innovation manager at a Big Four company, described how knowledge remains dispersed among many places:

> *"We see everywhere that people are constantly querying knowledge bases. So if, for example, you are a finance professional and a request comes in, you first have to assess it. You then consult eight different documents and six databases to look things up before ultimately giving an approval — yes or no. That is exactly the kind of work that can be greatly supported by RAG. However, people do not yet fully see that."*

This knowledge fragmentation creates inefficiencies beyond total search time. Employees must remember which systems contain which information, navigate different search mechanisms, and manually synthesise information scattered across various repositories.

P4, an AI consultant specialising in public sector and cybersecurity, explained how RAG enables document synthesis and cross-departmental knowledge access:

> *"It can give you a synthesis of multiple documents at once. Instead of me checking 6 separate documents, it will provide them all at once, which I can then check. Also, I can get knowledge from other domains more easily. So, for example, maybe there's a regulations team. I don't know the regulations, but I want to learn something small. With RAG, I don't need to find someone from regulations; I can just ask the system. I think this saves time and cost."*

P12 highlighted the ability to easily access knowledge scattered across multiple geographic locations, each with different systems. He stated:

> *"We built an application that brings together all of their sales data. They very often receive specific customer requests and then need to create a highly tailored proposal — for example, this needs to be transported to a certain location, this is the environment, this is the material required, and these are the associated risks. There are an enormous number of documents worldwide, literally thousands."*

The strategic silo-breaking motivation connects directly to the integrating KBDC capability; the recognising, transferring, and integrating of knowledge across organisational boundaries. P2 articulated this connection with a vision for the future:

> *"So we connected a human resources portal via RAG to our general-purpose chatbot. That is a first step. The final promise for RAG should be something comparable to enterprise search, where you are connected to the entire company."*

These findings demonstrate how RAG facilitates knowledge discovery across organisational boundaries, resulting in reduced information silos.

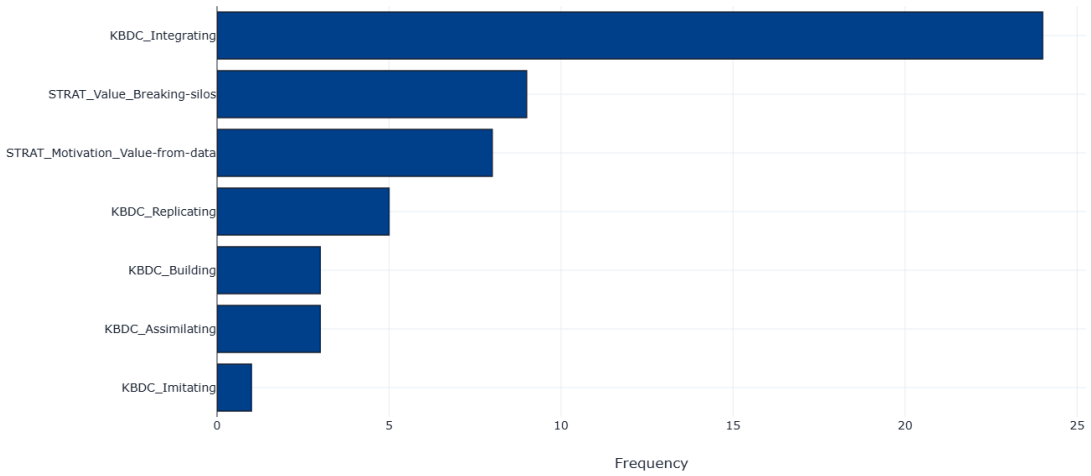

Figure 4.3: Frequency of codes within theme 1.2: RAG as a Means for Knowledge Discovery.

4.1.3 RAG as a Means for Creating Future AI Capabilities

Eight of 14 participants cited the development of internal AI capabilities as a strategic motivation for RAG adoption. These participants and their organisations viewed RAG not solely as a productivity tool or a knowledge-discovery mechanism, but also as a means of developing internal organisational AI expertise. Key codes for this theme are *STRAT_ Motivation_ AI_ Adoption*, *STRAT_ Learning_ Through _ Development*, *FUTURE_ RAG_ as_ Agentic_ Tool*, and *BUILD_ Pro_ Create_ AI_ Capability*. The code frequencies for theme 1.3 are shown in Figure 4.4.

The desire to create future AI capabilities was manifested in two ways: deliberate hands-on capability development and learning through technology-driven adoption.

Some organisations deliberately build RAG and GenAI capabilities internally by doing it themselves, creating AI expertise in the process. P3 articulated their independence strategy:

> *"Well, we think that AI will only work well when you combine it with your own data. That can be done in different ways, and RAG is a very important one. There are also other ways, like through MCP services and live retrieval. We have ideas about all of that, too. But the essence is that we want to get more value out of our data ourselves, and we want to do that ourselves. We do not want to leave it to external parties to do cool things with our data."*

P3 identified two clear advantages of keeping RAG development in-house: maintaining data control and developing organisational AI capabilities. Organisations that build RAG systems internally gain insight into implementation challenges, user behaviour, and technical limitations. P3 described this learning process as a competitive advantage over other universities:

> *"Because we do it ourselves, we therefore have our own chatbot platform for education. The data is ours, which allows us to see how chatbots are being used and learn from them. The biggest difference between us and most other educational institutions is that we are genuinely learning what works and what does not work."*

This learning-through-building approach was confirmed by other participants. P5, an AI developer at a Big Four company, reflected on the experience gained through development:

> *"We have gained a great deal of knowledge, not only through building it but also through using it. Because we developed everything ourselves, that knowledge also gradually permeated the organisation. As a result, there were many advantages."*

For some organisations, RAG's accessibility made it an attractive entry point for experimenting with emerging GenAI technologies. P11 described their exploratory adoption:

> *"RAG was something you read about occasionally, but within our organisation, it wasn't really a thing yet. Then we thought, okay, actually, this is a perfect use case for this new technology.*

Let's pioneer in this area, and then we will see how far we can get."

In contrast to deliberate capability development, some organisations adopt RAG via a technology push, in which technology adoption is driven by competitive pressure (Di Stefano et al., 2012). While this approach lacks the strategic clarity of deliberate capability development, it still creates opportunities for organisational learning. Participant P9, director of AI at a software vendor, observed this pattern at multiple clients:

> *"Organisations come to us with the question: we want to use GenAI, how do we do it? So often it is also a sort of a technology push, which is not necessarily a good thing."*

This GenAI technology push is consistent with the experiences of AI consultants, who have observed many instances of customers requesting AI solutions without clear use cases. P14, an AI consultant at a Big Four company, shared his experiences:

> *"Customers do not usually come to us directly with the problem for which they want RAG—that's already a bit too specific. Of course, they might say, 'We want our own chatbot,' and then naturally the next question from us is, 'What do you want to use it for?' If it turns out they want to use internal data, we do explain what RAG is, and then we can work together to create something and see if it works."*

Participant P7, an experienced AI consultant, elaborated on similar patterns in which clients sought to use GenAI, even though other solutions would have been better:

> *"Clients often come to us saying, 'We want this, and we want to use generative AI.' My response is usually: let's talk about the first part and forget the second for a moment. Generative AI is a means to an end, not an end goal in itself. We are not forcing a tool simply for the sake of using it. Sometimes that resonates well with clients, and sometimes it does not. In some cases, they insist on having generative AI included, for reasons that are not always entirely clear."*

Regardless of the adoption path, practitioners agreed that RAG is an easy entry point for organisations looking to introduce GenAI features beyond standard LLM licenses and procured general assistants. Whether adopted for specific capability development or in response to competitive pressure, RAG implementations enable organisations to develop internal GenAI expertise.

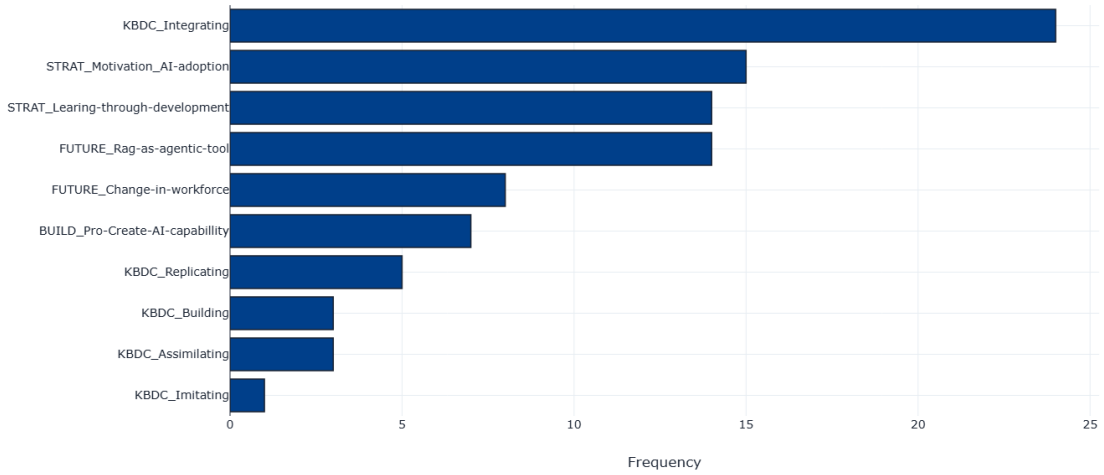

Figure 4.4: Frequency of codes within theme 1.3: RAG as a Means for Creating Future AI Capabilities

4.2 RAG Design Decisions

Organisations and practitioners encountered three key design decisions: (i) the build versus buy decision, (ii) the domain versus general scope decision, and (iii) the architecture complexity decision. Figure 4.5 shows the network for this theme, communicating which codes are related to it. The individual code labels have been removed for visual clarity. A heatmap illustrating the distribution of codes across interviews for this theme is provided in Appendix D, Figure D.2.

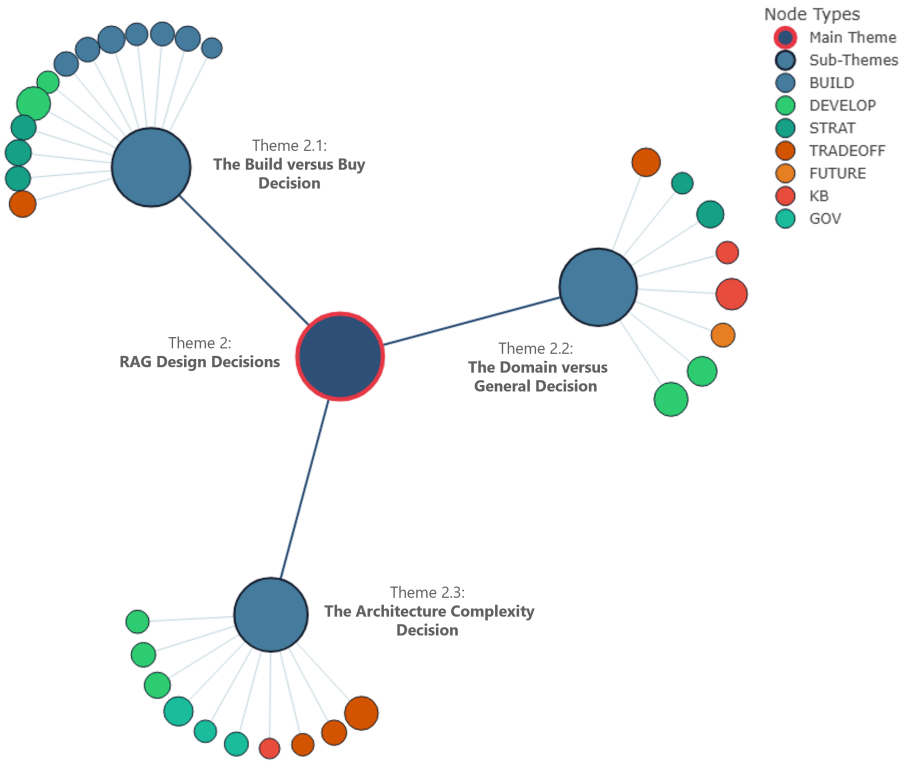

Figure 4.5: The code network for theme 2 and its three subthemes. Node size indicates relative code frequency.

4.2.1 The Build versus Buy Decision

All participants (14 of 14) discussed the build-versus-buy decision in RAG implementations. Rather than converging on a binary recommendation, participants identified a set of competing factors, including control, cost, time to market, IT capabilities, and strategic positioning, that organisations must weigh against their own context and requirements. Key codes for this theme were *DEVELOP_Successfactor_Business-Case*, *TRADEOFF_Build_vs_Buy*, *BUILD_Pro_Full_Ownership*, and *BUILD_Con_Need_IT_Capabilities*. Figure 4.6 shows the code frequencies for theme 2.1.

The Control and Governance Factor

Participant P2, an innovation manager at a Big Four company, discussed that their overall strategy is to build custom applications due to their high standards for control, governance, and security.
He articulated:

> *"We prefer to build solutions ourselves to maintain strong control and governance. With a clear overview of everything being developed, we can approve applications at the platform level, so only small components of individual solutions require separate approval, which accelerates our time to market. We aim to meet standards that vendor software alone cannot provide. While we sometimes use vendor platforms for speed, we prefer our own platform whenever possible, as much of it is already pre-approved, giving us greater control."*

This need for control was supported by P12, an AI consultant at the same company:

> *"Another disadvantage of vendor software is that it is a bit of a black box; you do not really know what is going on there."*

P11, a data scientist at a financial institution, further illustrated how the lack of control over vendor AI software can affect implementations. He described his experience during a parallel pilot using both a custom and a vendor system:

> *"At first, it may seem that purchased software performs much better, but as the use case develops, you start encountering various problems. That's when the advantages of a custom-built tool really become apparent. A non-custom solution often comes with all sorts of hidden issues that only emerge once you have spent a quarter working in depth to implement it."*

The Cost Factor

Practitioners also discussed the costs associated with the buy-versus-build decision. Participant P13, a data scientist at a Big Four company, noted there is a clear difference in cost between building it yourself with open source resources and procuring RAG from software vendors:

> *"I think RAG as a Service is extremely overpriced. Before we built our own solution, we had some sales discussions, and they priced us around $1.2 million for the RAG I was describing. When I looked into building it myself with open source resources, we estimated a cost of roughly $60,000. It is an insane difference. That said, I understand it can make sense for people who do not have the technical knowledge to build it in-house."*

P12 elaborated on the monetary trade-offs associated with building RAG in-house:

> *"Ultimately, you can build something yourself, which may be cheaper, but you need a team to maintain it. You need to fix bugs, update it, and so on. Alternatively, you can just purchase a solution. There are also trade-offs to consider: do you make a front-loaded investment that you recoup later, or spend the amount over a longer period?"*

The Time to Market Factor

Time-to-market emerged as an important factor in the build-versus-buy decision, with vendor software typically enabling faster deployment. Participant P12 explained that organisations must weigh the urgency of system deployment against long-term capability development:

> *"Time to market is another factor, how quickly do you want the solution to start adding value?"*

Participant P14, an AI consultant at a Big Four company, further elaborated on the resource trade-offs associated with developing RAG in-house:

> *"It depends, of course, on how quickly you need something. I think if you develop something in-house, the time to market is generally longer. Do you even have the expertise in-house, or would you need to hire new people?"*

However, P2 (see Control and Governance) noted that if governance and control requirements exceed what vendors can offer, it may actually be faster to build them in-house. As discussed earlier, their organisation maintains pre-approved custom platforms for application deployment, minimising the approval process.

The Additional Overhead Factor

Beyond initial development costs and time-to-market, practitioners also discussed the additional overhead required to run and maintain a RAG system. RAG demands continuous data updates and evaluation to ensure the application is working correctly. This requires dedicated IT resources, which is something not every organisation has.

Participant P5, an AI developer at a Big Four company, explained the lack of capacity and GenAI capabilities at organisations:

> *"An organisation's IT capacity is often dedicated to other purposes. For example, if you have an organisation focused on building semiconductors, they have many brilliant engineers and developers in-house. But I don't think they have a team at our level that can do what we do in terms of AI, so it is more efficient for them to acquire this capability from us. I genuinely believe that."*

Procuring vendor software transfers this operational burden to external providers. P14 illustrated the maintenance advantages of vendor solutions:

> *"The advantage of buying a solution is that you get it quickly, and it likely comes with an SLA, so it is maintained and updated. If you develop it yourself, you also need to maintain it and potentially expand your team whenever new, relevant techniques emerge."*

Participants agreed that the challenge for RAG lies not in development but in operations. Participant P9, director of AI at a software vendor, reinforced this notion by stating:

> *"You can indeed develop different RAGs very quickly for various domains, but you also need to be able to run them. That aspect is often overlooked. People tend to focus heavily on developing RAG, but the real challenge is being able to operate it, monitor it, and continuously improve upon it during operations."*

The Long-Term Viability Factor

A minority (2 of 14) raised concerns about whether their system would remain relevant in the near future, given the rapid evolution of AI. Participant P6, an ML engineer at a research infrastructure provider, noted that AI developments move so quickly that long development cycles can result in solutions that are already outdated by the time they are completed. He articulated that:

> *"That is a general problem in AI: building solutions that may become irrelevant by the time you are finished building them, because AI just moves so quickly."*

However, P6 did state that this does not justify avoiding custom development, but stressed that organisations need to be aware of whether what they are building will actually provide long-term value. P11 reinforced this concern by warning against building generic solutions that could become embedded in popular ecosystems:

> *"If you start developing something custom, it is also important to avoid building a generic solution yourself. That is an easy trap to fall into, trying to create something that all teams can use. In doing so, you often end up building functionality that Microsoft will probably release as well, making your entire product redundant."*

These five factors do not operate in isolation, but interact dynamically based on organisational context. Participants noted that organisations should weigh these factors against their own strategy to determine which are more important. In general, participants agreed that if organisations possess the IT capacity to build their own RAG, they should do so if it will create sufficient value.

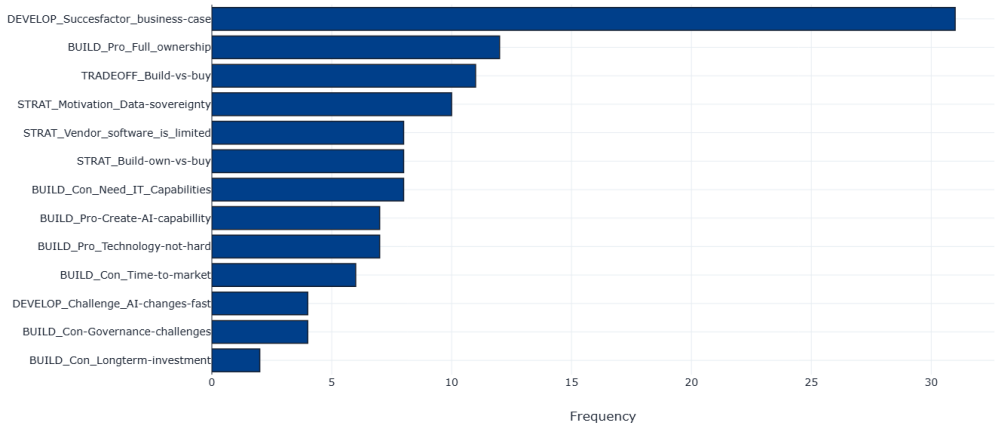

Figure 4.6: Frequency of codes within theme 2.1: The Build versus Buy Decision.

4.2.2 The Domain versus General Decision

A critical design decision in RAG implementations concerns the choice between building (i) horizontal, broadly scoped RAG systems that can be used across multiple domains or entire organisations, and (ii) vertical, domain-specific RAG systems that deliver targeted value to specific teams or departments. Key codes for this theme were *DEVELOP_Successfactor_Business-Case*, *KB_Scope_Domain-specific*, *DEVELOP_Successfactor_Start-Small*, and *TRADEOFF_General-vs-Domain-Applicability*.
Figure 4.7 shows the code frequencies for theme 2.2.

Organisations that pursued horizontal RAGs encountered challenges around scope management and data relevance. P5, who successfully deployed a horizontal RAG application, explained the constant pressure to expand their solution scope:

> *"We receive many proposals to add extra datasets to our horizontally deployed RAG, and then it becomes a data problem. The core question for us is: how do we ensure that the data not only stays up to date, but also remains relevant for the entire organisation?"*

This relevance challenge manifests concretely when departments request additions that only serve narrow user groups. As P5 elaborated:

> *"For example, a department may want to include specific regulations and standards. While these are used intensively by a small group, the majority of the organisation does not care about them. Offering such data at a global level would therefore have limited value."*

Without clear boundaries, horizontal RAG initiatives risk becoming bloated and losing strategic focus. Participant P7, an experienced AI consultant, described a failed project he worked on:

> *"We had constantly changing directions. Initially, we wanted to hyper-focus on one scenario, then shifted to another, and later decided to build a baseline that could serve all cases. This eventually expanded into a broader foundation, leading to a full platform. The lack of a clear vision meant we never reached a true endpoint; it became a continuous process of building, building, and building."*

In contrast, participants working on domain-specific RAG applications (11 of 14) reported fewer problems regarding scope creep and consistently advocated for this approach over horizontal deployments. P9 described the optimal RAG strategy as building a generic framework and reusable infrastructure for domain-specific deployments:

> *"I think a generic approach to build domain-specific RAGs is the best. You start with a specific problem and then attach a specific corpus to it. So if it's HR, you use a corpus of HR documents; if it's self-service, you use a corpus of self-service documents. A generic approach allows you to build these domain-specific RAGs quickly, rather than throwing everything into one large pool, which does not work. It is much more effective to define separate categories and domain-specific solutions."*

Participant P11 captured the fundamental risk of horizontal deployment in one sentence:

> *"If you start with a generic solution scope, you will often get a solution that becomes one size fits none instead of one size fits all."*

Across interviews, participants converged on a consistent pattern: organisations that started with domain-specific RAG were better positioned to expand incrementally if needed, whereas horizontally scoped RAGs faced constant stakeholder pressure to include more data in the system.

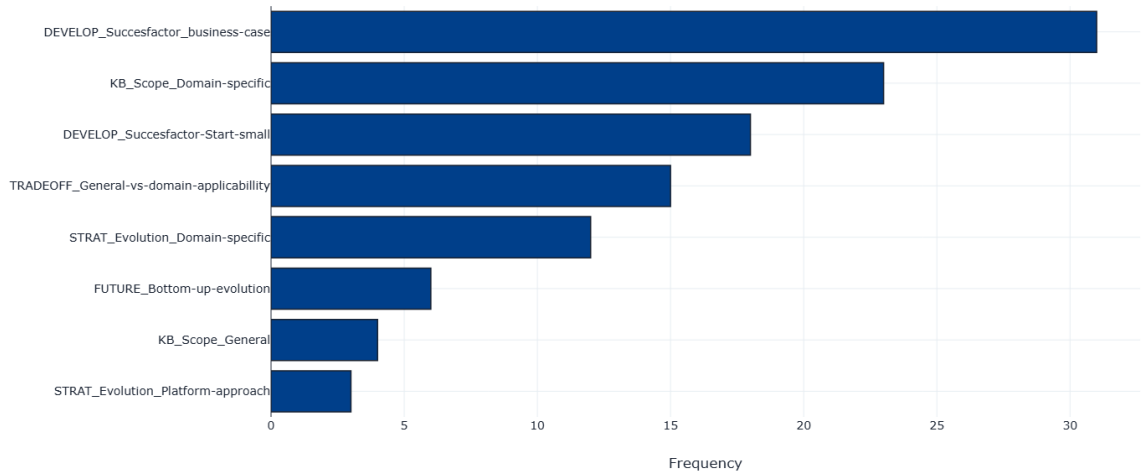

Figure 4.7: Frequency of codes within theme 2.2: The Domain versus General Decision.

4.2.3 The Architecture Complexity Decision

We discussed RAG architecture with 11 out of 14 participants, ranging from detailed technical explanations with developers and AI engineers to more high-level descriptions with managers and consultants. While we established in our literature review that many novel variations of RAG exist, practitioners generally favoured simple vector-based advanced RAG architectures (see Section 2.3) over agentic and graph-based approaches, as only two of 14 participants had built operational Agentic RAGs. Key codes for this theme were *TRADEOFF_Simplicity_vs_Performance*, *DEVELOP_Successfactor_Scalable*, *TRADEOFF_Deploymentspeed_vs_Perfection*, and *DEVELOP_Successfactor_Pragmatic_Development*. Figure 4.8 shows the code frequencies across theme 2.3.

Participant P7 identified overengineering as a common pitfall in RAG development:

> *"Pilots get too complicated themselves. And then you have overzealous engineers who really hyper-tune it. When you hyper-tune it, it works on one kind of data that you already know, but then you get a slight deviation from that kind of data and see that you overfitted."*

Beyond the risks of overfitting, participants also highlighted trade-offs introduced by more complex architectures, particularly agentic approaches. P12 explained that:

> *"There is always a trade-off with agentic approaches. If you add a reflection step, it increases quality—there is scientific evidence for that. But it also means it takes longer to reach an answer. So the user experience does not necessarily improve, and it becomes more expensive. Thus, is it really necessary for every application to do it that way?"*

Several practitioners explained that complex architectures and heavy optimisation often result in diminishing returns. Participant P11 described how this influenced their design choices:

> *"We use a fairly standard RAG architecture and have not deviated much from established patterns. We found that raw model performance matters less than expected; users primarily care about the user experience and the interface. As a result, we focused on improving responsiveness by reducing latency. As an ML engineer or data scientist, you are used to optimising models, but with GenAI, that does not matter as much."*

Architectural considerations proved especially important for organisations operating at scale. Participant P6, an ML engineer, explained how architectural complexity directly impacted latency due to the size of their knowledge base (1.5 terabytes):

> *"I would describe our architecture as very simple. We built the bare minimum. Initially, we tried a much more complex approach that leveraged all available indexing features, but it was extremely slow, with searches taking almost five minutes. We eventually switched to a simple, direct indexing approach, which reduced search time to three to five seconds. The performance trade-off was minimal."*

These lessons led practitioners to advocate for a more incremental, pragmatic approach, in which solutions start small and simple. P11 articulated:

> *"The most successful generative AI use cases I see do not start too broadly. They focus on optimising a single, specific task first, and only expand more broadly after that."*

P7 further elaborated on why starting small enables future scaling:

> *"You need to start small and simple, then slowly expand the solution scope. Start with one single use case, then add bits and pieces to it and realign. This is better than hyper-tuning and over-engineering one specific use case, since you will realise that if you want to really prove value, you are going to have to do 10 different use cases. To do that, you build a common baseline, and then you end up having done a lot of overengineering on the original use case. So that doesn't work well."*

Across interviews, practitioners consistently emphasised that starting small is essential for RAG success. Despite the sophisticated techniques documented in academic literature, production systems tended toward simpler architectures that prioritised user experience and maintainability over technical optimisation. They argued that organisations should assess whether complex architectures are actually needed to maximise value delivery, rather than defaulting to complex approaches.

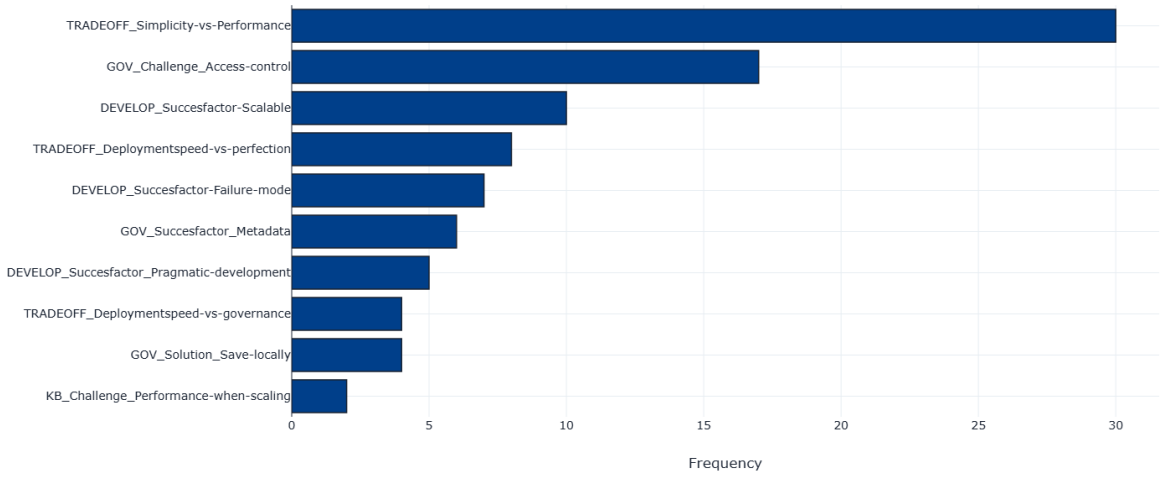

Figure 4.8: Frequency of codes within theme 2.3: The Architecture Complexity Decision.

4.3 Constraints in RAG Implementations

Organisations and practitioners encountered five primary constraints in RAG implementations: (i) data quality and ownership, (ii) governance and control, (iii) the business case and expectation management, (iv) RAG evaluation, and (v) RAG adoption challenges. Figure 4.9 shows the network for this theme, communicating which codes are related to it. The individual code labels have been removed for visual clarity. A heatmap illustrating the distribution of codes across interviews for this theme is provided in Appendix D, Figure D.3.

Figure 4.9: The code network for theme three and its five subthemes. Node size indicates relative code frequency.

The following sections analyse the subthemes shown in Figure 4.9.

4.3.1 Data Quality and Ownership

Data quality and data ownership emerged as critical constraints for RAG implementations, with 12 of the 14 participants discussing them. These constraints reflect organisational practices related to documentation standards, data maintenance, and ownership structures. Participants consistently noted that poor or outdated data cannot be compensated for by a more sophisticated architecture. The key codes for this theme were *DATAQUAL_Successfactor_Standard_Format*, *DATAQUAL_Challenge_Format-inconsistent*, *DATAQUAL_Successfactor_Ownership_Decentralized*, and *KB_Challenge_Data_Staleness*. Figure 4.10 shows the code frequencies for theme 3.1.

The Format Challenge

Organisational data is often inconsistent, reflecting issues such as scanned documents, varying file formats, and missing information. Participant P7, an AI consultant at a Big Four company, explained the data issues he experienced during one of his projects.

"I have encountered situations where I was given PDF documents that were scanned copies of printed materials. In some cases, OCR allows us to extract the text, but then it turns out that some documents are very poorly scanned or have an unusual structure. Others are laid out in complex tabular formats, which means you have to build additional processing around them."

Similar challenges related to data quality and format consistency were also reported by P12, an AI consultant at a Big Four company:

"There was a huge problem with data quality. They had a SharePoint structure with many folders, but those folders were not built with the same structure. There were different versions of the same document in there. These were factors that made it very difficult to find what you were looking for."

Participant P1, an AI Engineer at a semiconductor company, further illustrated format issues by describing the guidelines he provided when colleagues wanted to ingest new files:

"If you have a Word version of the file, then just upload the Word version. Do not upload a PDF file because there's a lot of nonsense... I would also suggest that they write a document that is semi-structured, so as to use identifiers, sections or titles to split the document correctly, instead of a bunch of non-styled, non-formatting text."

The Freshness Challenge

In addition to challenges with data formatting and consistency, participants also described issues with data freshness. Participant P5, an internal AI developer at a Big Four company, described freshness as a critical factor for RAG success, emphasising its importance over technical sophistication. P5 illustrated:

"Ultimately, the biggest problem you face is not an AI problem, but a data maintenance problem: how do you keep the data up to date? That is essentially a classic information management challenge."

Some participants (4 out of 14) explicitly noted that they used automatic polling to track file changes, updating the knowledge base whenever files were modified. However, even if the knowledge base is updated automatically, it does not guarantee that the data is always fresh. Participant P13, a data scientist at a Big Four company, illustrated this limitation:

"There's a RAG that we worked on that scrapes new data sources every day. But there are always hiccups in that process... some people are going to expect this document that just came out needs to be in there already. But in practice, data has got to get scraped, preprocessed, sent to some database, and then needs to be indexed. There's a whole set of things, and it can actually take a while before it's fully indexed and retrievable."

Participant P14, also an AI consultant at a Big Four company, explained that organisations should first become more mature in data management capabilities before starting RAG:

"Companies often do not know what data they have and who should be able to see which data... They have 100 different versions of documents scattered around, not knowing which is the most current, and have a whole lot of data quality issues that need to be solved before you can even start with RAG."

The Ownership Challenge

Participant P7 provided particularly detailed insights into data ownership, drawing on extensive hands-on experience with RAG implementations. While other participants mentioned data ownership, they did not elaborate on the mechanisms or organisational practices in the same depth. P7 explained that data staleness is a direct result of unclear data ownership, as employees do not want to be responsible for maintaining data and often defer to others:

"In order to get RAG success, somebody needs to own the data, maintain it, but then as we see with any organisation, documentation is something nobody wants to touch."

He further illustrated this issue with an example from one of his projects, where the lack of data ownership caused practical problems.

> *"We also found, when speaking with different people, that the data was often not up to date. Information was not maintained, or another source had become the source of truth within the company, while the master document was never updated. As a result, no one really took ownership. When we pointed this out and asked a department to fix it, the response was often that the responsible person was no longer with the company."*

To address this challenge, P7 emphasised the need for a robust governance model that defines clear responsibilities and standards:

> *"You need a very strong governance model... You need to set expectations right from the beginning, but also set responsibilities for maintaining data and give them a standard for data quality. You need to define: this is what it needs to be, this is how it needs to be structured."*

P7 advocated centralised governance with decentralised data ownership as a promising solution to the data ownership challenge. It ensures that individual teams maintain their own data while adhering to overarching standards and formats, creating a unified data landscape.

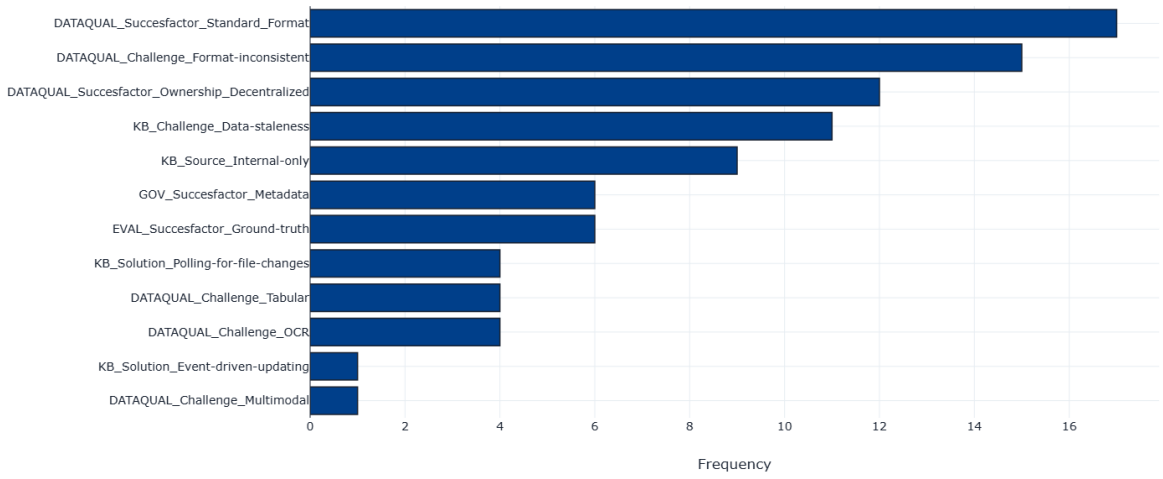

Figure 4.10: Frequency of codes within theme 3.1: Data Quality and Ownership.

4.3.2 Governance and Control

Participants identified governance and control as a further organisational constraint in RAG implementations, particularly regarding access control, compliance, and data residency. The key codes for this theme were *GOV_Challenge_Access_Control*, *GOV_Challenge_Compliance*, *STRAT_Motivation_Risk_Mitigation*, and *GOV_Challenge_Data_Residency*. Figure 4.11 shows the code frequencies for theme 3.2.

Participant P14 explained how access control is a critical element of enterprise RAG:

> *"I think you especially run into the problems that companies don't really know what data they have and who should be able to see which data. You naturally want to prevent that with RAG suddenly everyone can access everything. You will need to perform metadata filtering based on the currently logged-in user, making certain documents available or not. And that is a question that we as consultants cannot answer; our client has to figure that out themselves."*

The importance of robust access control was confirmed by Participant P3, a product owner at a research institution, describing it as their biggest challenge yet to solve:

> *"I think governance is the most challenging aspect: ensuring that we can expose the right data to the right people, without it costing us too much time or effort, and doing it in a way that is as automated as possible."*

Access control becomes even more challenging when scaling across multiple teams, as illustrated by P11:

> *"If we expand to multiple teams, it becomes more difficult because then we have to separate documents from each other... You could also build different vector databases, but I don't think that scales. If you have 2-3 teams, that's an option, but not with 10. And we hope to roll it out further. It's easier if everything is in one vector database."*

In addition to access control, participants highlighted concerns related to data residency and the use of organisational data in the training of new LLMs:

> *People are generally very cautious. Users are afraid that if they implement RAG, their data will go everywhere: will it comply with regulations, who will be able to access it, and will models be trained on their data? Those concerns definitely exist.*

Participant P5 addressed these concerns by saving all conversational history locally on the user's laptop, thereby avoiding storage overhead. P5 described his solution as:

> *"Every time you start a new chat, I write a small file to your local computer. Instead of maintaining a central database, I decided that if we are not allowed to store anything ourselves, then all the data that already exists on your computer is apparently allowed to be there. In that case, I might as well write it locally to your machine... When chats need to be removed due to GDPR requirements, for example, when someone leaves the organisation, you can simply wipe that computer remotely."*

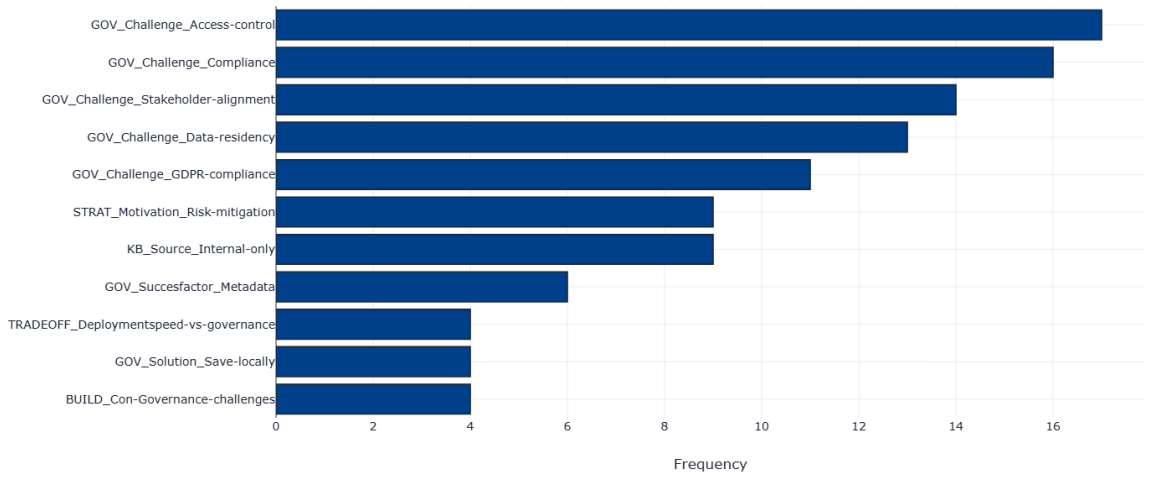

Figure 4.11: Frequency of codes within theme 3.2: Governance and Control.

4.3.3 The Business Case and Expectation Management

To build an RAG application, developers must interact with multiple stakeholders, including the business, risk, IT, and management. These stakeholders need to be convinced that the envisioned RAG system will create enough value to justify its existence. A critical element of the justification is a strong business case, with 12 of 14 participants discussing aspects of it. Practitioners also emphasised that managing stakeholder expectations throughout development is essential to maintain support and prevent scope creep. The key codes for this theme were:
DEVELOP_Successfactor_Business_Case, *GOV_Challenge_Stakeholder_Alignment*, *ADOPT_Success factor_Manage_Expectations*, and *DEVELOP_Challenge_Define_Success*. Figure 4.12 shows the code frequencies in theme 3.3.

Participant P8 noted that business cases in Gen AI are often not fully grounded:

> *"I have seen a lot of business cases that were based on rough estimates. But it remains hard if you do not possess the data to back up your claims. And sometimes you do not know if you will get a return on investment, since there are many factors, such as data quality, that will determine value creation"*

Without a clear business case, it will be harder to convince stakeholders that the RAG application will actually deliver value, increasing the chance of the project being cancelled. Participant P11 described

how he experienced the many Gen AI projects that were cancelled:

> *"Our organisation was very strict in which GenAI use cases were allowed to continue. There were all kinds of requirements, and many use cases were declined. All teams wanted to build GenAI use cases, but many had no clear business value."*

Another problem with GenAI and RAG is that demos and proofs of concept are easy to set up and can deliver impressive results. However, this often leads non-technical stakeholders to overestimate GenAI's capabilities, believing it can solve any problem. As Participant P7 noted, after a successful proof of concept with a small, curated knowledge base, performance dropped when scaling to production. He noted:

> *You need to set expectations right from the beginning... What I often see is that if you raise a data quality issue at the very end, saying 'oh by the way, the solution is not performing very well because your data is not great', the client starts thinking we have not done our job well and blames us.*

Participants noted that most people do not really understand AI. They frequently view it as a silver bullet that can solve all kinds of problems. Participant P11 described this as:

> *"Expectation management is very important, because people often see AI as some kind of wonder... I think you're actually better off sketching somewhat lower expectations, so that they become more enthusiastic over time."*

Participant P14 described a similar pattern, where stakeholders do not understand what GenAI can and cannot do, highlighting the importance of expectation management:

> *"Sometimes we give a demo, and people are looking at us like they see water burning. And because many people come from zero, they think it can do anything. They do not understand the limitations, and it is on us to explain: this looks nice and flashy, but in these circumstances it does not work."*

Practitioners converged on the importance of expectation management, highlighting the need inform stakeholders about limitations, requirements and expected value creation. These expectation management challenges connect directly to the data quality and governance issues discussed in earlier sections. If stakeholders are not informed early of these constraints, even technically sound RAG systems risk being perceived as failures in real-world operational environments.

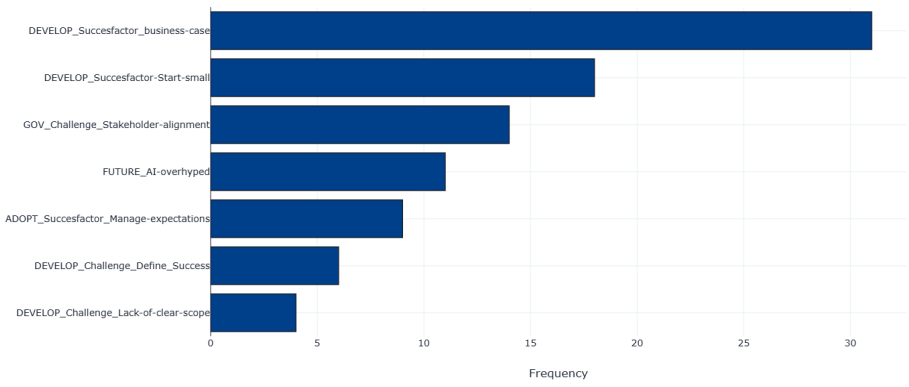

Figure 4.12: Frequency of codes within theme 3.3: The Business Case and Expectation Management.

4.3.4 RAG Evaluation

A strong business case sets expectations for what a RAG system should deliver and defines the required value it should create. In practice, it is difficult to determine whether the system meets those expectations. Unlike traditional software, RAG outputs are non-deterministic, subjective, and context-dependent. While RAG can be evaluated, it requires constant monitoring and will never be perfect. While evaluation did not arise in every interview (8 out of 14), practitioners who did mention it emphasised its

importance in operational settings. Key codes for this theme were *EVAL_ Challenge_ Which_ Metric_ To-Use*, *EVAL_ Successfactor_ Automated_ Evaluation*, *DEVELOP_ Successfactor_ Scalable*, and *EVAL_ Successfactor_ Large_ Scale_ Testing*. Figure 4.13 shows the code frequencies within theme 3.4.

Participant P7, an AI consultant, noted that his clients do not always know how to define RAG success, which ties back to their GenAI technology push. He stated:

> *"In cases of AI or RAG, clients do not know what a good solution looks like in some cases. And it is a bit frustrating as well... So success criteria is also something that we usually have to define, and that ties back to: what is our experience? What do we think success should be, and how do we make sure we reach that success?"*

Participant P14 further elaborated on the evaluation challenge when scaling beyond proof of concepts, highlighting the need for end-to-end pipeline evaluation to monitor real-time usage:

> *"When building proof of concepts for clients, we can demo it ourselves. We ask the questions we know the system is suited for. But in production, you get the craziest questions: multi-hop reasoning, infinite turns, people suddenly changing topics. You will not encounter that in a proof of concept because it is a controlled environment. In production, you will run into problems. And if you do not properly evaluate how your pipeline handles those situations, you will definitely get issues."*

As systems scale, evaluation increasingly relies on automated methods, which introduces additional limitations. One common approach is to use LLMs to evaluate the outputs of other LLMs. Participant P4 pointed out its inherent problem:

> *"With LLMs, usually you cannot really measure performance, there is no perfect metric that you can use. With RAG frameworks, you can prepare a ground truth file... but it is still LLMs checking LLMs, so it is not perfect."*

Because of these limitations, participants emphasised that RAG evaluation cannot rely solely on automated metrics, but also requires human feedback. Participant P7 described how they apply user testing for system evaluation:

> *"First of all, the quality of results. We almost always do a user test – we give it to a few users, saying 'please use it, ask questions and tell us how it's performing.'... We ask testers to judge on six parameters: Are the answers correct? Is it in acceptable language? Is it formatted well? Is it easy to understand? Do answers contain all the information? And are they completely based on the documents provided?"*

Participant P9 confirmed the need for human-feedback, but explained the difficulties in getting detailed feedback from regular users:

> *"The feedback systems inside your RAG application are important, but I want to note that features such as giving thumbs up and down are limited. Only a small percentage of people use this, and only when they are very satisfied or dissatisfied. So you need more explicit feedback about what went wrong."*

P9 further elaborated on the need for large-scale RAG evaluation. He argued that RAG, with its use of LLMs, will never answer all questions correctly and that developers should accept that. They should not worry about individual questions; instead, they should evaluate RAG performance across a large set of questions. He explained that:

> *"If you have a proper large-scale automated evaluation, you can say: it's unfortunate that a single question does not perform well, but when we look at ten thousand questions overall, the performance is good. It does not have to be perfect; it is acceptable if an individual query fails"*

Collectively, these findings reveal evaluation as a constraint for RAG implementations. Defining success is inherently ambiguous, technical metrics are imperfect, and collecting human feedback requires substantial effort. Organisations that underestimate the need for continuous evaluation risk deploying systems that cannot be adequately monitored or improved, thereby undermining the value proposition of their original business case.

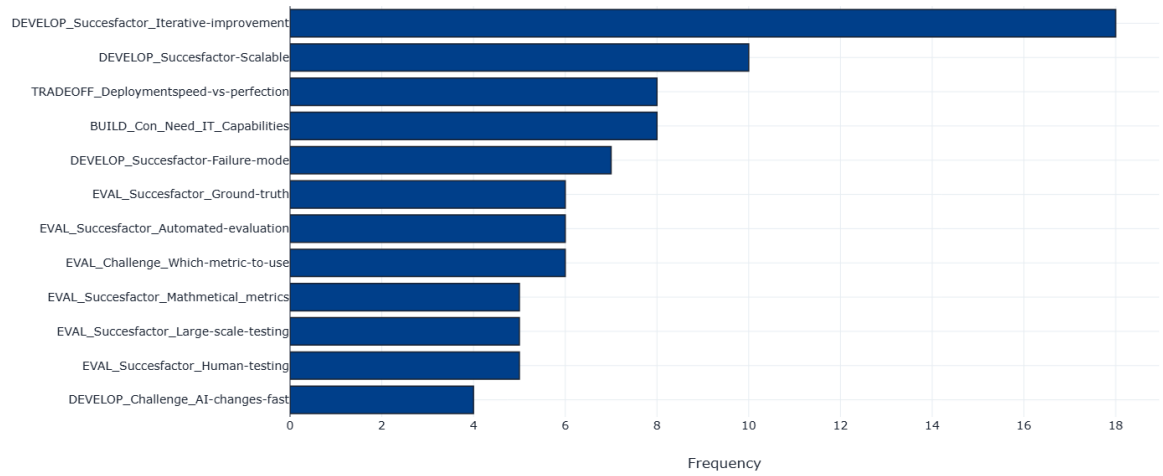

Figure 4.13: Frequency of codes within theme 3.4: RAG Evaluation.

4.3.5 Adoption Challenges

A RAG system cannot be considered successful by mathematical evaluation metrics alone; it requires sustained usage from its intended user base. Adoption emerged as a critical factor, with 14 out of 14 participants actively discussing the subject. Key codes for this theme were: *ADOPT_ Successfactor_ Human _ Feedback*, *ADOPT_ Challenge_ Lack_ Of_ Training*, *ADOPT_ Successfactor_ Change_ Management*, and *ADOPT_ Successfactor_ User_ Experience*. Figure 4.14 shows the code frequencies within theme 3.5.

When asked how they define RAG success, Participant P12 emphasised actual deployment and use rather than technical performance alone:

> *"What do you define as RAG success? I would take it one step further: that it goes into production and that it is actually used by people."*

Participant P2 emphasised that adoption accelerates once users experience value in their daily work. He reflected on the rollout of an internal general-purpose GenAI tool:

> *"An effect of doing it right is that adoption goes fast. When value is visible, you will see adoption follow. We did a minimal launch, and people picked it up immediately."*

Multiple participants discussed methods to demonstrate the value of RAG systems. P4 explained that gaining user trust is essential, noting that negative early experiences can lead to abandonment:

> *"If users get a couple of answers that are not satisfactory, they will stop using it. And if the system is only able to answer very simple questions, why even use it? You need to show them what it can mean for their job and give hands-on exercises."*

This need for trust building directly ties back to the importance of expectation management, highlighting the need for users to know system capabilities and limitations. Participant P11 explained how user education was used to support adoption during the launch of their RAG application:

> *"At the very beginning, when we went live, we created a presentation of around twenty slides with various do's and don'ts. For example: do not ask two questions in the same message, phrase your questions in a specific way, and similar guidelines. The idea was that users also need to learn how to use such an application properly, and that this guidance could serve as support for them."*

The need for clear communication was also emphasised by participant P3, highlighting that uncertainty and fear around AI usage can suppress adoption even when tools are readily available.

> *"The most important thing for AI adoption in education is communication. There is a lot of fear around students using AI for their homework, but I think that problem is much smaller than portrayed. A bigger problem is that students are actually afraid to use AI because they*

fear being accused of plagiarism... If you do not accompany AI tools with clear communication explaining how and why to use them, adoption gets stuck at 10%."

These findings indicate that simply making a RAG system available is insufficient to ensure adoption. Participant P2, an innovation manager, framed adoption primarily as a human problem:

"How do you do change management? How do you deal with fears and concerns? There is definitely a soft side to this – more so than the technical side at the moment."

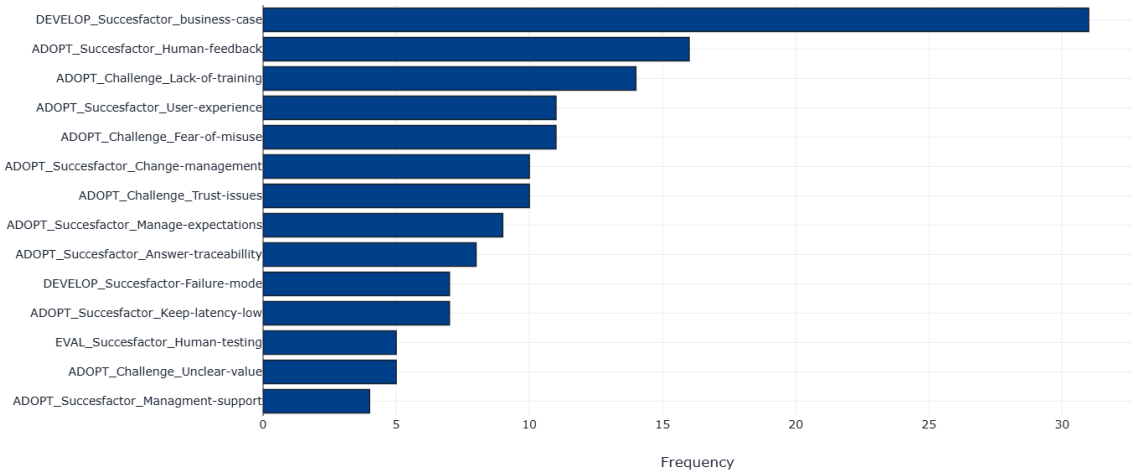

Figure 4.14: Frequency of codes within theme 3.5: Adoption Challenges.

Chapter 5

Discussion

This chapter interprets the main findings from our thematic analysis to answer the research question stated in Section 1.2. We situate our findings within the literature discussed in Chapter 2 to create an empirical understanding of enterprise RAG. Findings are integrated into a framework that explains how the identified factors interact dynamically. From this synthesis, we propose 11 design principles grounded in practitioner insights. Finally, this chapter reflects on the limitations of our research and its implications.

5.1 Empirical Understanding of Enterprise RAG

This section synthesises our empirical findings across the three themes identified in Chapter 4 to develop an integrated RAG understanding. We will not reiterate thematic content; rather, we will focus on how strategic considerations, foundational design decisions, operational design decisions, adoption enablers, and constraints interact in real-world enterprise implementations, highlighting where our findings diverge from or converge with existing literature.

Strategic Considerations
Strategic considerations reflect the reasons why organisations adopt RAG. We found that organisations pursue RAG for three main reasons: (i) improving productivity, (ii) improving knowledge discovery, and (iii) creating future AI capabilities. We introduced KBDC in Chapter 2 as our primary lens for analysing the strategic role of RAG, utilising a simplified typology adapted from Denford (2013). We established that RAG can theoretically enable integrating, replicating, building, assimilating and imitating. However, our empirical findings reveal a gap between these theoretical capabilities and actual organisational usage.

The productivity and knowledge discovery motivations both align particularly well with the integrating capability. Integrating is the process of recognising internal knowledge sources, facilitating their transfer, and integrating them to create value. Productivity improvements occur when RAG helps employees quickly integrate information from multiple sources, and knowledge discovery occurs when RAG reveals previously unknown information across siloed knowledge domains.

The third motivation involved the use of RAG to create internal AI capabilities. Here, RAG did not merely serve as a knowledge tool but also as an entry point for organisational use of GenAI. This learning-by-doing approach reflects that the process of building RAG is considered strategically valuable. While the strategy of developing future AI capabilities does not fit within the KBDC framework, resulting systems still enable productivity and/or knowledge discovery.

Across interviews, most organisations were not yet actively pursuing other KBDC capabilities. We identified replicating in only a few cases. For example, RAG implementations serving as global knowledge platforms intended to capture and transfer expertise. However, such strategic knowledge management applications proved rare compared to productivity-focused implementations. None of the participants explicitly mentioned having functionalities that align with building capabilities (creating new knowledge by recombining existing knowledge); such capabilities appeared only in discussions of future possibilities.

External capabilities proved challenging due to governance constraints, with participants emphasising the significance of exposing data to the internet. While one participant did enable secure RAG internet access, he argued this was only possible due to his organisation's partnership with Microsoft.

We argue that this focus on integrating partly reflects the immaturity of Agentic RAG, which is better suited to enable more complex capabilities. Moreover, many systems were just entering production and were based on early GenAI use cases. As RAG and agents become more mature, organisations should seek functionality that goes beyond simple document question answering, exploring other KBDC capabilities to realise RAG's full potential.

Several participants, especially those who worked as consultants, explained that some organisations adopt RAG via a technology push. This pattern suggests that organisations adopt RAG under competitive pressure, rather than in response to clearly articulated technology demand. This empirical pattern was confirmed by a recent industry survey, in which executives expressed concern about falling behind, reflecting the importance of competitive pressure in GenAI adoption (Deloitte, 2025b). Due to this technology-push approach, there is often no clear business case or defined set of success metrics, reducing the likelihood of fully capturing the value of RAG. Similarly, these organisations sometimes pursued GenAI solutions when a standard workflow or classical machine learning model would have been more appropriate. This technology-push pattern exemplifies the epistemic risk discussed in our literature review (see Section 2.2.4), in which design decisions misalign due to limited RAG understanding.

Foundational Design Decisions

Foundational design decisions represent specific choices that relate to the strategic intent, turning strategy into system capabilities. They determine what the RAG system can do and create path dependencies which constrain future options; see Section 2.2.4. We identified six key decisions based on our literature review and empirical patterns found across interviews.

When adopting RAG, the first choice organisations face is the envisioned solution scope. It shapes, among other factors, the boundaries of the knowledge base, governance complexity, and feasible architectural options. Practitioners strongly agreed that a domain-limited (vertical) solution is the best approach, as it can create targeted value and is easier to operate. Even participants who built general-purpose RAGs advocated for a domain-limited approach. RAGs excel when tuned to specific datasets and problems, but become less useful and more expensive as the amount of data increases. With domain-specific RAG solutions, people can ask specific and complicated questions about their daily work, whereas general-purpose RAGs focus on covering general topics across the organisation, such as HR. This pattern was also confirmed by the scientific literature, with multiple case studies emphasising the importance of keeping RAG domain limited (Brehme et al., 2025; Hasan et al., 2025).

Similarly, organisations need to decide whether to develop custom RAGs or procure vendor software. Practitioners strongly favoured custom development, but noted that this is not a universal answer and depends on factors such as available IT capacity and required time-to-market. Practitioners preferred custom development because it afforded them full control, enabled the development of AI capabilities, and was often less expensive overall. Furthermore, vendor software often has numerous limitations that reduce usability for more complex, production-ready use cases. This preference aligns with KBDC theory (Zollo and Winter, 2002; Denford, 2013), where developing internal RAG capabilities for knowledge management enables organisations to create capabilities that serve as sources of sustained competitive advantage. However, practitioners agreed that building custom RAG requires investments and long-term commitment, signalling that organisations need to have a clear strategy before starting development.

Another recurring insight is that practitioners clearly preferred simple vector-based RAG approaches, as they argued that these solutions offered satisfactory performance. They often refer to their architectures as standard industry practice, reflecting their preference for systems that are easy to understand and maintain. In practice, most of the systems encountered would be categorised as falling under the Advanced RAG paradigm (Y. Gao et al., 2024). Although a small number of participants had some experience with Agentic RAG, they reported that it is unnecessary for many use cases. Similarly, graph-based approaches were generally perceived as overly complex by practitioners, but were recognised for their value in specific use cases in which relations among data points are essential. This pattern contrasts with the increasingly sophisticated architectures proposed in scientific literature (Asai et al., 2023;

Peng et al., 2024; Singh et al., 2025), as practitioners consistently prioritised maintainability over limited performance gains.

Access control design represents who can access which information via the RAG system. Operational RAG systems will typically contain documents with various access control levels. Practitioners agreed that, without robust access control systems, RAG solutions cannot scale safely across multiple teams or domains, thereby limiting future scalability. The literature also recognises access control as a critical factor in RAG system design, emphasising the need for robust governance and policy alignment (Akkiraju et al., 2024; Brehme et al., 2025). Access control design thus also represents a scalability decision, as organisations do not only need to assess what types of access control methods and policies are required now, but also those in the potential future. However, access control design is heavily dependent on the maturity of existing organisational governance, as operational RAG systems mostly inherit existing policies and controls. If an organisation already has access control issues, RAG will only exacerbate them.

From a system architecture perspective, the infrastructure architecture determines scalability, costs, and compliance capabilities. It refers to the underlying computational environment in which RAG systems operate, including the deployment model (cloud, on-premises, hybrid) and the supporting services. While often partly constrained by existing IT policies, regulatory requirements, and legacy systems rather than actively chosen (OECD, 2025), some degree of choice remains. Multiple participants reported using cloud-deployed LLMs for their RAGs, noting that operational costs were manageable because token prices have decreased substantially over the years. However, token consumption is rising due to increased AI usage and larger context windows. To address this, some organisations are shifting components of GenAI pipelines from public cloud providers to hybrid deployments, in which inference is performed on locally deployed models (Deloitte, 2025a).

Another key empirical pattern identified is the need for clear success metrics. This requires defining measurable success objectives before development begins. Organisations must explain beforehand how RAG creates value and, in turn, how this value will be measured. This allows developers to better align design choices with the envisioned system but also serves as a basis for evaluating whether the system achieves its original strategic goal. However, practitioners agreed that this is not an easy task, particularly when organisations have little prior experience with GenAI.

Compared with the technology-push adoption of GenAI and RAG, this lack of success metrics helps explain why many GenAI pilots fail to progress beyond proof of concept stages. In technology-push contexts, organisations often lack a clearly defined use case or success metric, making it difficult to assess whether the system delivers meaningful value. Without explicit criteria for success, pilots cannot be evaluated properly, design trade-offs remain arbitrary, and organisational support weakens over time. As a result, even technically functional RAG systems risk being abandoned because their contribution to the organisation and its goals remains unclear.

Operational Design Decisions

In our literature review, we defined operational design decisions as those that optimise performance within the technical and organisational boundaries established by foundational decisions. Examples of such decisions include LLM selection, chunking strategies, and system prompts. These decisions thus primarily concern the design of the steps within the RAG pipeline itself. Importantly, operational decisions are relatively reversible: LLMs can be replaced, chunk sizes increased, and prompts modified without fundamentally altering system capabilities. In Chapter 2, we found that the literature is primarily focused on these operational design decisions, aiming to optimise RAG performance (Y. Gao et al., 2024; Oche et al., 2025; Hasan et al., 2025).

Practitioners articulated that many developers focus on optimising these operational design choices, whereas in enterprise environments, this is often unnecessary. For example, they often spend considerable time engineering system instructions, including the handling of multiple edge cases and fallback answers. In practice, simple prompts often perform well enough and are easier to maintain or modify. Furthermore, regular users do not notice a small increase in answer performance; they are more interested in a smooth user experience, which shifts the focus from pipeline optimisation towards UI and UX development (Kim et al., 2025). In addition, heavy optimisation makes RAG systems less scalable, as they are fine-tuned for a single specific goal. While such optimisation may be justified when a system delivers sub-

stantial value, it creates systems that are hard to maintain, extend, or refactor when organisations evolve.

We thus identified a common trap in RAG: professionals accustomed to intensive model optimisation in traditional machine learning often unnecessarily apply the same approach to GenAI. Simple prompts frequently perform adequately and remain easier to maintain; basic chunking strategies usually suffice; simple hybrid searches can retrieve the correct chunks. Based on our findings, we argue that a more effective approach is to deploy a baseline RAG with limited optimisation and iteratively refine it based on user feedback, thereby reducing over-engineering and ensuring alignment with user needs.

Adoption Enablers

The adoption enablers determine if the RAG system actually delivers value, as even technically sophisticated systems fail if no one uses them in their daily work. The Technology Acceptance Model (TAM) states that technology adoption depends on two key perceptions: perceived usefulness and perceived ease of use (Davis, 1989). Our findings revealed how these factors specifically manifest in RAG implementations.

Perceived usefulness refers to users' belief that a system enhances their job performance. In the RAG context, systems demonstrate utility when they enable KBDC capabilities, which are directly tied to their ability to deliver relevant, reliable answers. This highlights the importance of educating users in using the system, enabling them to maximise its application in their daily work. Practitioners emphasised the importance of educating users on how to use RAG and GenAI, as these technologies are inherently error-prone (hallucinations). If users cannot understand why or when these mistakes occur, they will discard the system because they no longer trust its output. Similarly, users need to know what the system can and cannot do, as RAG is not a silver bullet that can solve every user problem.

Perceived ease of use reflects users' perceived effort to operate the system. While many users are already familiar with external GenAI tools such as ChatGPT or Gemini, internal RAG applications face additional challenges. Due to this familiarity, users are less tolerant of low speed and clunky interfaces (Challapally et al., 2025). Furthermore, RAG requires more advanced GenAI literacy, as users must understand prompting, failure modes, limitations, and answer verification.

These TAM factors cannot be addressed through technical design alone. Consistent with Ettinger (2025), we found that RAG adoption requires organisational change management. One participant even stated that RAG is no longer a technological problem but a human one. For years, employees have performed manual document searches, which will be partially replaced by RAG and AI agents in the near future. RAG represents the first generation of GenAI applications that go beyond simple general-purpose chatbots, initiating the transition to AI-powered workflows.

Constraints

Constraints determine whether and how strategic intent will be realised within an organisation. Drawing on the Technology-Organisation-Environment (TOE) framework (Tornatzky et al., 1990), we found that organisational constraints outweigh technical ones, despite RAG being a technological GenAI design pattern. Participants agreed that RAG technology is not particularly complex, which we also experienced during the development of our RAG prototypes. However, participants clearly noted that the move from proof of concept to production has multiple critical constraints. These constraints continuously influence RAG designs and implementations and help explain why organisations with similar strategic goals create different RAG systems. Based on our literature review and the empirical patterns we found, we identified six key constraints for enterprise RAG.

Data Quality and Ownership - Organisational

Data quality and ownership emerged as the primary organisational constraints to RAG's success. Participants agreed that if you have bad data quality–for example, due to various formats, data staleness, or bad version control–your RAG system will not perform well, regardless of the sophisticated RAG methods you might use. While small-scale proofs of concept can be corrected manually, this approach cannot be scaled sustainably to larger use cases. Data quality thus represents an organisational capability that determines the effectiveness of production-ready RAG systems.

The data ownership challenge reflects why improving data management is difficult. Participants noted that people do not want to handle documentation or be responsible for specific files. In practice, this means that people do not take responsibility for maintaining files, leading to the data issues described before. Improving data quality and data ownership requires large transformations, especially for organisations lacking maturity in data management practices (Bernardo et al., 2024). Therefore, data quality and ownership constrain RAG: organisations with poor data quality will remain limited to small-scale RAG implementations.

Governance and Control - Organisational
Governance and control emerged as the second organisational constraint for RAG. RAG systems should retrieve only information that users are authorised to access, based on their role/clearance. RAG depends on organisational governance practices and uses existing access control policies for data retrieval. In the absence of mature governance and access-control policies, the adoption of RAG may amplify existing organisational risks, given its ability to quickly retrieve vast amounts of confidential data via natural-language search (Akkiraju et al., 2024; Bruckhaus, 2024). Similarly, modifying existing governance and access control policies constitutes a transformation in itself, adding substantial overhead. Governance and control thus constitute a critical constraint for RAG, with organisations lacking governance and control maturity being constrained to limited RAG deployment and custom access control policies.

Stakeholder Management - Organisational
The organisational constraint not encountered in the RAG literature, but fully revealed by interviews, is stakeholder management. Participants noted that RAG implementations require coordination among a multitude of stakeholders, including technical teams, business teams, the risk department, and management. These stakeholders have varying priorities and needs, increasing the risk of scope creep and bloated systems. The number of stakeholders is partly dependent on the solution scope, as a horizontal general-purpose system has multiple business units competing for prioritisation.

Participants noted the importance of securing managerial support early, which improved alignment with other stakeholders, which aligns with literature on the importance of managerial support in AI adoption (Soomro et al., 2025). Furthermore, RAG requires strong project management to ensure stakeholder buy-in. Organisations that lack project and stakeholder management capabilities may see their RAG implementation fail due to protracted debates over features, knowledge base scope, risk, and governance.

Evaluation Challenges - Technological
While organisational constraints dominate enterprise RAG, evaluation emerged as a critical technical constraint. Participants struggled to evaluate their RAG systems, not always knowing how to measure pipeline performance. Mathematical metrics can measure elements such as retrieval accuracy, but cannot capture whether an answer was actually useful. While gold-standard references often exist for some questions, they cannot address all queries. Similarly, human evaluation does not scale to production-ready RAG systems. While the LLM-as-a-judge method scales, it relies on GenAI checking GenAI, and thus remains vulnerable to hallucinations. These evaluation challenges also stemmed from literature case studies, in which authors noted the lack of universal RAG measures (Xu et al., 2025; Brehme et al., 2025).

These evaluation challenges constrain operational optimisation. Development teams need reliable metrics to assess the impact of changes on the system and to determine whether success criteria are met. This lack of universal RAG metrics leads organisations to rely on human feedback and adoption metrics as proxies for system quality (Hasan et al., 2025). While this is not inherently wrong, it would be preferable to use large-scale automated metrics that fully capture end-to-end system behaviour.

IT Capabilities - Technological
IT capabilities, capacity, and technological maturity form explicit technological constraints. While not particularly complex, RAG requires some knowledge of machine learning, GenAI, and system operations. Without these, practitioners agreed organisations should likely not opt to build RAG themselves; instead, they should rely on vendor software. Similarly, organisations with limited maturity should not begin by building agentic RAG, as poorly implemented agents could cause substantial harm within the organisation (Narajala and Narayan, 2025). However, for large organisations with dedicated IT departments and adequate expertise, implementing RAG is feasible and generally manageable. Thus, while the IT capabilities constrain RAG implementations, they should not be the primary consideration when adopting RAG.

Compliance and Regulation - Environment
Organisations implementing RAG must comply with applicable regulations, such as GDPR and HIPAA. They constrain what can be ingested into the knowledge base, the types of questions that can be asked, and how and where conversational data should be stored. Regulations can vary by sector and organisation, underscoring the need for developers to collaborate with risk teams to determine the solution's scope. Organisations with mature IT and Risk departments will be able to address regulatory issues more readily. In contrast, organisations with immature compliance capabilities will face more challenges when trying to scale RAG across the organisation. While the literature acknowledges the challenges of compliance and regulation (Xu et al., 2025; Oche et al., 2025), our findings suggest that compliance constraints compound with governance immaturity, underscoring its importance for RAG.

Overall, these empirical patterns indicate that enterprise RAG success is primarily determined by organisational readiness rather than pipeline complexity. Technical sophistication alone cannot compensate for weak data foundations, an unclear strategy, or insufficient support for adoption.

5.2 Factors Influencing RAG Implementations: a Framework

Our research aimed to examine what factors practitioners faced when implementing RAG in enterprise environments. We interviewed 14 practitioners, who varied in roles, organisations, and solution scope. We stated the following research question:

Research Question. *What factors influence Retrieval-Augmented Generation (RAG) implementations in large organisations?*

Our findings indicate that RAG implementations are influenced by multiple factors that interact dynamically: strategic intent shapes foundational design decisions, which establish boundaries that operational decisions optimise within, while adoption factors validate alignment, and constraints filter what system is feasible throughout this process. RAG is thus more than a technical pattern; to successfully deploy it, organisations need to align choices across the different factors. Based on our empirical understanding (Section 5.1), we propose an integrated framework (see Figure 5.1) that explains how different factors in RAG implementations interact. The framework expands on the theoretical integration shown in Figure 2.3 in Chapter 2. We show an overview of key empirical findings per framework dimension in Appendix B, Table B.1.

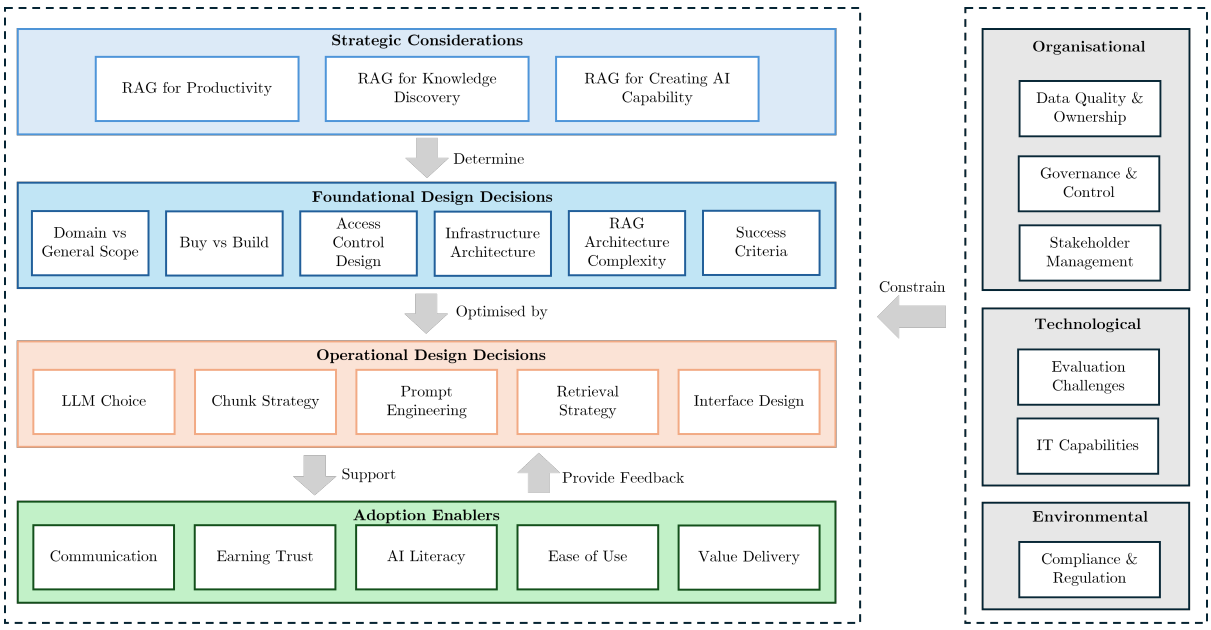

Figure 5.1: Our framework shows how RAG is shaped by strategic considerations, foundational design choices, operational design choices, adoption enablers, and constraints.

As shown in the framework, RAG implementations are shaped by four interconnected layers: strategic considerations, foundational design choices, operational design choices, and adoption enablers. These four layers are continuously influenced by constraints, being either technological, organisational, or environmental (Tornatzky et al., 1990).

Our findings indicate that RAG success requires alignment across all layers: strategic intent must be translated into foundational design decisions, the resulting system must be optimised for the task through operational decisions, and adoption enablers must ensure the system's value is visible. When misalignment occurs between layers, such as when an organisation pursues enterprise-wide RAG ambitions while having immature governance policies, implementations are less likely to achieve their original goals despite technical sophistication.

Foundational decisions establish the system's boundaries, which are then optimised by operational decisions. For example, a simple vector-based system requires different pipeline elements than a complex multi-agent one. While the operational choices shown in our framework are not exhaustive, they reflect that these choices are technical optimisations rather than system boundaries.

When people use the system, they create a feedback loop that can be used to optimise operational design decisions to better match user needs. It is the responsibility of the development team to interact with users and stakeholders to gather in-person feedback, as feedback mechanisms embedded in RAG systems are often limited due to their lack of contextual richness. Critically, persistent low adoption rates could also signal more fundamental problems, such as the knowledge base not aligning with user information needs, poorly defined success criteria, or overall strategic misalignment due to vague goals at the start of the project. Adoption metrics, therefore, serve as a continuous validator for the alignment across all layers in our proposed framework.

Constraints act as filters that determine which paths remain viable, but their binding nature depends on choices across different layers. For example, IT capabilities become more relevant when trying to build complex agentic systems, or compliance and regulation might require more attention in sectors such as healthcare and HR. Constraints thus interact with decisions rather than simply blocking paths.

The interactions within our framework explain why organisations that pursue RAG for similar strategic purposes may follow very different solution paths, as there is no universally optimal approach. Rather, it relies on the alignment between strategic goals, implementation, and contextual constraints.

5.2.1 Implementation Pitfalls

While our framework shows how alignment across layers enables RAG success, it also reveals how misalignment leads to failure. We identified four key implementation pitfalls: strategic ambiguity, over-engineering, adoption neglect, and constraint blindness. These pitfalls demonstrate that RAG failures stem from organisational and strategic misalignment rather than technical pipeline performance.

Strategic Ambiguity
When organisations adopt RAG through a technology-push approach, they often lack robust use cases. This produces strategic ambiguity for what the system should be able to do, which cascades through the foundational design decisions. The resulting systems are more likely to contain decisions that do not align, creating problems that cannot be resolved by simply optimising the system.

Over-Engineering RAG Pipelines
Over-engineering operational decisions yields diminishing returns, as users rarely notice small improvements in returned answers. Developers trained in traditional machine learning tend to over-optimise RAG pipelines when simple configurations suffice. The more complex your architecture, the more you can customise and tune, but this entails trade-offs in cost, maintainability, and scalability.

Adoption as Afterthought
Technically sophisticated systems fail when adoption enablers are neglected. Developers should aim to involve users in the early stages of development, research their problems, and examine how RAG can solve them. Making a system available is not enough, as RAG requires at least some understanding of AI to be used effectively. Users should be educated and continuously involved, not merely be recipients

of another system.

Constraint Blindness
Organisations may pursue ambitious designs while fundamentally misunderstanding their own constraints. For example, organisations may pursue enterprise-wide RAG despite immature data governance or the construction of complex agentic systems without the required AI capabilities. This creates a clear misalignment between strategic intent and the realistic organisational possibilities. To resolve this misalignment, they should either revise their goals or enhance the capabilities required to realise the original vision.

5.3 RAG Design Principles

Having established how factors interact within our framework and how misalignment manifests as various implementation pitfalls, we now translate these insights into actionable guidance through 11 RAG design principles, as shown in Table 5.1. They represent analytically derived enterprise RAG recommendations grounded in practitioner experiences.

Principles 1-2 address strategic considerations, and principles 3-4 concern the constraints layer. Principles 5-7 guide foundational design decisions. Principles 8-9 address operational design decisions, and principle 10 focuses on the adoption layer. Principle 11 addresses horizontal scaling, which requires the correct execution of all other design principles.

The principles are intended for practitioners and decision-makers in RAG, including ML engineers, data scientists, management, and business stakeholders. The design principles act both as a guide and a tool to create a common RAG understanding, thereby aligning various stakeholders on the best approaches.

They also reflect that successful RAG implementation shares multiple characteristics with traditional software engineering as clear requirements, user-friendly design, and organisational alignment remain critical.

Table 5.1: RAG design principles for enterprise implementations.

ID	Design Principle	Rationale
1	Define strategy before building	Identify target KBDC capabilities and avoid technology-push adoption. Clear strategic intent prevents misalignment across design layers
2	Create clear business case	Define measurable objectives with actual calculations rather than rough estimates. Clear metrics enable value measurement and guide development decisions
3	Assess organisational readiness	Assess technological, organisational, and environmental constraints honestly to prevent overambitious designs
4	Prioritise organisational over technical	Data quality and governance require change management, not just technical fixes. Architectural sophistication cannot compensate for organisational immaturity
5	Start domain-specific	Domain-focused implementations deliver higher value with more manageable complexity than general-purpose ones
6	Design access control early	Access control maturity determines scalability beyond pilots. Design and integrate mechanisms from the start, even in small implementations
7	Keep architecture simple	Simple vector-based systems often achieve satisfactory performance and have a lower maintenance burden. Agentic approaches should be justified by their use case
8	Avoid over-engineering operational decisions	Operational optimisation yields diminishing returns, as users rarely notice marginal performance improvements.
9	Iterate based on actual usage	Deploy baseline systems quickly, then improve based on actual usage rather than assumptions

Continued on next page

Table 5.1 – continued from previous page

ID	Design Principle	Rationale
10	Embed adoption into design	Technical excellence without user adoption delivers no organisational value. Integrate adoption enablers from the start
11	Scale horizontally when mature	Start with a domain implementation, then scale RAG horizontally as a modular building block once proven. Modular design enables reuse across domains and integration with other (agentic) systems

5.4 Threats to Validity

While our research provides valuable insights into RAG implementations, we acknowledge several limitations that flow from our methodology and research choices (see Chapter 3). We examine our research by using the four criteria for qualitative research: credibility, dependability, confirmability, and transferability (Guba and Lincoln, 1985). Given the exploratory and qualitative nature of our research, the findings do not establish causal relationships; rather, they capture patterns and practitioners' experiences.

Credibility

Our research faced credibility threats due to potential confirmation bias and subjective interpretation. Given the researchers' backgrounds in computer science and machine learning, RAG was initially approached from an optimisation perspective. We addressed this through our reflexive method, allowing us to recognise and shift from this bias after early interviews revealed the importance of contextual factors. Coding was conducted iteratively, allowing emerging code groups and themes to be refined as the analysis progressed. Furthermore, we reached theoretical saturation after interview 8, allowing subsequent interviews to serve a confirmatory role. Nevertheless, alternative interpretations remain possible, and interpretive bias cannot be fully mitigated.

Dependability

Dependability is affected by both procedural and temporal factors. First, the use of semi-structured interviews and the variety of participant roles led to conversations that varied in depth and topic. While the interview guide proved useful for covering core topics, researchers often deviated from it to better explore participants' experiences. Second, although iterative coding ensured that the overarching elements of the interviews were consistently captured, we employed a single, evolving coding framework. Earlier coding iterations were deleted or revised, thereby limiting traceability over time.

Another threat to dependability arises from the rapid evolution of RAG and GenAI. The literature review was conducted between September 2025 and November 2025, with expert interviews taking place between October 2025 and December 2025. Because GenAI and RAG evolve rapidly, our research provides a snapshot of RAG's development. During our research period, RAG deployment became increasingly accessible, with companies such as OpenAI and Anthropic integrating small-scale RAG features directly into their LLM offerings. Furthermore, our research took place at the peak of GenAI hype, which likely inflated the technology-push pattern. While we are confident that our framework remains valid in the future, it remains to be seen how commoditised RAG will become.

Confirmability

As the research was conducted by a single researcher, inherent threats to confirmability arise. Both code and theme construction are subject to interpretation, and other researchers might have identified different results. In particular, the abstraction of interviews into high-level themes and the construction of the framework and design principles require substantial interpretive judgement.

We maintained reflexivity throughout the research process by continuously questioning our assumptions. In addition, interpretations were iteratively checked against academic and grey literature to ensure that findings did not reflect only the researcher's own perspective.

Transferability

Our sampling strategy achieved diversity across roles, sectors, and system scopes, improving transferability to large organisations. However, our sampling strategy restricts generalisability in multiple ways.

First, all participants and organisations operated within the Dutch and European regulatory environment, particularly the GDPR compliance requirements, which differ across geographical locations. Therefore, our results may not fully capture the challenges across regions.

Second, our research focused on large organisations in the Netherlands (over 250 employees). Therefore, our findings may not generalise to SMEs, as they have different requirements and IT capacity. For example, in a small company with little IT knowledge, IT capabilities might be a more binding constraint. While we estimate that our framework remains largely applicable, we have not validated it in SME environments.

Third, 9 of 14 participants worked for the same Big Four consulting company, potentially skewing the results with their organisational perspective of RAG. However, these consultants also provided breadth across multiple client organisations, thereby contributing to the diversity of examined RAG systems.

Last, we focused on internal RAG applications from a knowledge management perspective. While some participants worked on customer-facing RAGs, we did not explore their specific challenges and requirements, such as real-time performance demands, public-facing risk management, or handling adversarial queries. Therefore, our findings primarily reflect RAG in internal knowledge management contexts.

Despite these threats, our research achieved saturation across diverse roles and organisations. The patterns we identified align with those found in scientific and grey literature, supporting the credibility and transferability of our proposed framework and design principles.

Chapter 6

Conclusions

6.1 Answer to the Research Question

This research investigated what factors influence RAG implementations in large organisations (>250 employees). We conducted an inductive, reflexive thematic analysis of 14 semi-structured interviews, with participants representing diverse roles, organisations, and system scopes. Our methodology combined prototyping, an extensive literature review, and thematic analysis to develop empirical insights grounded in practical experiences rather than theoretical possibilities. This resulted in three distinct themes which we synthesised into an empirical understanding centred around key elements of RAG (strategy, design, adoption, and constraints). While our research question was descriptive in nature, our findings enabled us to extend beyond description by developing an explanatory framework and prescriptive design principles.

We synthesised our findings into a framework that reveals how RAG implementations are shaped by four interconnected layers: Strategic Considerations, Foundational Design Choices, Operational Design Choices, and Adoption Enablers. These four layers are continuously constrained by technological, organisational, and environmental factors. We found that organisations primarily pursue RAG to improve productivity and knowledge discovery, aligning with the integrating KBDC capability. Furthermore, practitioners consistently favoured simple vector-based systems over complex approaches, revealing a practitioner-literature gap where academic focus on sophisticated architectures misaligns with current enterprise needs. Critically, organisational constraints (data quality, governance maturity, stakeholder management) emerged as more binding than technical and environmental constraints.

Success in enterprise RAG depends primarily on organisational readiness rather than pipeline complexity. While RAG proofs of concept can be built rapidly, scaling beyond limited use cases requires strategic clarity, governance maturity, and consistent data formats. Our research revealed that most organisations are still in the early stages of applying RAG to knowledge management, as more advanced KBDC capabilities remain unexplored. As RAG continues to evolve and mature, a lasting competitive advantage will not emerge from architectural sophistication, but from investments in required organisational capabilities.

6.2 Contributions

This thesis makes both academic and practical contributions to the field of Retrieval-Augmented Generation. Although our research question was descriptive, our approach also yielded prescriptive contributions.

6.2.1 Academic Contributions

This research contributes to the RAG literature by taking a holistic approach. Existing literature focuses on (i) novel architectures, (ii) optimisation, (iii) evaluation, (iv) simple use cases, or (v) Agentic RAG. Our approach combined these separate topics by engaging with practitioners in various roles who build enterprise RAG. We developed an empirical understanding which we then synthesised into a framework

that explains differences in RAG implementations across organisations. Our framework integrated the KBDC typology (Denford, 2013), the TOE framework (Tornatzky et al., 1990), and TAM framework (Davis, 1989) to show how a multitude of factors shape RAG implementations.

We extended the KBDC theory to RAG, which, to the best of our knowledge, has not been explored before. We showed that RAG can primarily be applied to internal KBDC capabilities, but, in practice, only integrating is actively pursued. We predict that this will change as RAG and GenAI mature within organisations.

With our research, we have partially closed the theory versus practitioner gap by investigating on what actually happens in the RAG enterprise context. While literature focuses on sophisticated architectures and optimisation methods, practitioners see these as common RAG pitfalls. We found that the primary challenges revolve around organisational readiness and strategic alignment, rather than technical optimisation.

Last, we distinguished between foundational and operational design decisions, a separation not explicitly addressed in existing RAG literature. Foundational design decisions (scope, architecture, access control, infrastructure) create boundaries and path dependencies that determine system capabilities. Operational decisions (LLM choice, chunking strategy, prompting) can only optimise the system that lies within these boundaries. This separation explains why practitioners who focus on operational optimisation rather than foundational alignment will encounter substantial challenges when scaling within an organisation.

6.2.2 Practical Contributions

Due to its holistic and practitioner-focused nature, our research makes multiple practical contributions. First, practitioners can use our framework to better assess their own RAG readiness. Organisations can identify key constraints early and address these before starting RAG development. Conversely, if key identified constraints cannot be addressed, organisations gain a realistic understanding that scaling RAG across their enterprise will remain infeasible, regardless of their architectural sophistication.

Second, we proposed 11 design principles that capture RAG best practices. These principles provide actionable guidance to help practitioners navigate and avoid common pitfalls, such as over-engineering and neglecting user needs.

Third, our findings contrast with the constant hype around GenAI. Organisations do not require niche or agentic architectures, as simple solutions often deliver satisfactory performance. While there is value in Agentic RAG for specific use cases that require complex reasoning, organisations should not fall into the technology-push trap. We emphasise that organisations should first identify RAG use cases, then adopt the technology, not the other way around. This does not mean that organisations should wait to improve required capabilities, such as infrastructure or data quality, but rather that they should avoid endless GenAI prototyping without clear goals or strategy.

6.3 Future Work

Enterprise RAG offers multiple opportunities for future work. First, our research included many early-stage RAG systems, providing insights into only a part of their software lifecycle. Future work should track RAG systems throughout their lifecycles to investigate how RAG evolves and to what extent foundational decisions truly create binding path dependencies. This research could also address RAG maturity by examining stage characteristics, enablers, and show-stoppers that determine progression from pilot to successful vertical and/or horizontal deployment.

We concluded that organisational readiness determines RAG success, and although we developed a framework to identify key factors, we provided no prescriptive readiness framework. Future work should create readiness frameworks for various use cases and RAG system paradigms that help organisations to better assess their current situation.

Furthermore, the role of RAG will likely shift as organisations and agentic AI mature. RAG will likely

no longer be a stand-alone application, but rather a tool that autonomous agents call to retrieve specific information. This raises many additional questions that we did not address in our research. Should organisations deploy a single centralised agent with multiple sub-agents, or multiple domain-limited agents with full access within their respective domains? How do governance and access control principles translate from stand-alone RAG to agent-orchestrated retrieval? Future work should investigate the role of RAG in an agentic enterprise and how existing RAGs could be applied for agentic usage.

In addition, the emergence of internal RAG tooling that enables employees to build their own small RAGs warrants investigation. Moreover, general-purpose assistants such as Microsoft Copilot and Google's Gemini are increasingly offering more functionality, raising questions about how these tools will coexist in the future. In addition to these general-purpose assistants, software vendors are increasingly embedding GenAI functionality into their own systems, leading to a siloed experience. Future research should investigate how these systems will interact and which role custom RAG implementations will play in an ecosystem likely dominated by GenAI assistants and internal vendor-provided GenAI features.

Bibliography

Adams, G. L., & Lamont, B. T. (2003). Knowledge management systems and developing sustainable competitive advantage. *Journal of Knowledge Management*, *7*(2), 142–154. https://doi.org/10.1108/13673270310477342

Akkiraju, R., Xu, A., Bora, D., Yu, T., An, L., Seth, V., Shukla, A., Gundecha, P., Mehta, H., Jha, A., Raj, P., Balasubramanian, A., Maram, M., Muthusamy, G., Annepally, S. R., Knowles, S., Du, M., Burnett, N., Javiya, S., … Boitano, J. (2024). Facts about building retrieval augmented generation-based chatbots. http://arxiv.org/abs/2407.07858

Al Yami, M., Ajmal, M. M., & Balasubramanian, S. (2021). Does size matter? the effects of public sector organizational size' on knowledge management processes and operational efficiency. *VINE Journal of Information and Knowledge Management Systems*, *52*(5), 670–700. https://doi.org/10.1108/VJIKMS-07-2020-0123

Alavi, M., & Leidner, D. (2001). Review: Knowledge management and knowledge management systems: Conceptual foundations and research issues. *MIS quarterly*, *1*, 107–. https://doi.org/10.2307/3250961

Alavi, M., & Leidner, D. E. (1999). Knowledge management systems: Issues, challenges, and benefits. *Communications of the Association for Information Systems*, *1*(1), Article 7. https://doi.org/10.17705/1CAIS.00107

Ammann, L., Ott, S., Landolt, C. R., & Lehmann, M. P. (2025). Securing rag: A risk assessment and mitigation framework. https://arxiv.org/abs/2505.08728

An, Y., Cheng, Y., Park, S. J., & Jiang, J. (2025). Hyperrag: Enhancing quality-efficiency tradeoffs in retrieval-augmented generation with reranker kv-cache reuse. http://arxiv.org/abs/2504.02921

Asai, A., Wu, Z., Wang, Y., Sil, A., & Hajishirzi, H. (2023). Self-rag: Learning to retrieve, generate, and critique through self-reflection. http://arxiv.org/abs/2310.11511

Balaguer, A., Benara, V., de Freitas Cunha, R. L., de M. Estevão Filho, R., Hendry, T., Holstein, D., Marsman, J., Mecklenburg, N., Malvar, S., Nunes, L. O., Padilha, R., Sharp, M., Silva, B., Sharma, S., Aski, V., & Chandra, R. (2024). Rag vs fine-tuning: Pipelines, tradeoffs, and a case study on agriculture. http://arxiv.org/abs/2401.08406

Barnett, S., Kurniawan, S., Thudumu, S., Brannelly, Z., & Abdelrazek, M. (2024). Seven failure points when engineering a retrieval augmented generation system. *Proceedings - 2024 IEEE/ACM 3rd International Conference on AI Engineering - Software Engineering for AI, CAIN 2024*, 194–199. https://doi.org/10.1145/3644815.3644945

Barney, J. (1986). Strategic factor markets: Expectations, luck, and business strategy. *Management Science*, *32*(10), 1231–1241. http://www.jstor.org/stable/2631697

Barney, J. (1991). Firm resources and sustained competitive advantage. *Journal of Management*, *17*, 99–120. https://doi.org/10.1177/014920639101700108

Barreto, I. (2010). Dynamic capabilities: A review of past research and an agenda for the future. *Journal of Management*, *36*, 256–280. https://doi.org/10.1177/0149206309350776

Bernardo, B. M. V., Mamede, H. S., Barroso, J. M. P., & dos Santos, V. M. P. D. (2024). Data governance quality management—innovation and breakthroughs across different fields. *Journal of Innovation Knowledge*, *9*(4), 100598. https://doi.org/https://doi.org/10.1016/j.jik.2024.100598

Bjarnason, E., Lang, F., & Mjöberg, A. (2023). An empirically based model of software prototyping: A mapping study and a multi-case study. *Empirical Software Engineering*, *28*. https://doi.org/10.1007/s10664-023-10331-w

Bouabdallah, A., Gavilan, J., Gerbl, J., & Patumcharoenpol, P. (2021). Multimodal approach for metadata extraction from german scientific publications. https://arxiv.org/abs/2111.05736

Braun, V., & Clarke, V. (2006). Using thematic analysis in psychology. *Qualitative Research in Psychology*, *3*(2), 77–101. https://doi.org/10.1191/1478088706qp063oa

Brehme, L., Dornauer, B., Ströhle, T., Ehrhart, M., & Breu, R. (2025). Retrieval-augmented generation in industry: An interview study on use cases, requirements, challenges, and evaluation. https://arxiv.org/abs/2508.14066

Bruckhaus, T. (2024). Rag does not work for enterprises. https://arxiv.org/abs/2406.04369

Challapally, A., Pease, C., Raskar, R., & Chari, P. (2025). *The genai divide: State of ai in business 2025* (Retrieved 30 November, 2025). MIT NANDA. https://mlq.ai/media/quarterly_decks/v0.1_State_of_AI_in_Business_2025_Report.pdf

Cheng, M., Luo, Y., Ouyang, J., Liu, Q., Liu, H., Li, L., Yu, S., Zhang, B., Cao, J., Ma, J., Wang, D., & Chen, E. (2025). *A survey on knowledge-oriented retrieval-augmented generation* (tech. rep.). https://doi.org/https://doi.org/10.48550/arXiv.2503.10677

Conner, K. R. (1991). A historical comparison of resource-based theory and five schools of thought within industrial organization economics: Do we have a new theory of the firm? *Journal of Management*, *17*(1), 121–154. https://doi.org/10.1177/014920639101700109

Dalkir, K. (2011). *Knowledge management in theory and practice*. The MIT Press. Retrieved September 11, 2025, from http://www.jstor.org/stable/j.ctt5hhhx9

Davenport, T. H., & Prusak, L. (2000). Working knowledge: How organizations manage what they know. *Ubiquity*, *2000*(August). https://doi.org/10.1145/347634.348775

Davis, F. D. (1989). Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS Quarterly: Management Information Systems*, *13*, 319–339. https://doi.org/10.2307/249008

Deloitte. (2025a). Tech trends 2026. https://mkto.deloitte.com/rs/712-CNF-326/images/DI_Tech-trends-2026.pdf

Deloitte. (2025b, October). *Ai roi: The paradox of rising investment and elusive returns* (tech. rep.) (Retrieved on 16 January, 2026). https://www.deloitte.com/content/dam/assets-zone2/nl/en/docs/issues/generative-ai/deloitte-nl-gen-ai-ai-roi-survey.pdf

Denford, J. S. (2013, March). Building knowledge: Developing a knowledge-based dynamic capabilities typology. https://doi.org/10.1108/13673271311315150

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding. http://arxiv.org/abs/1810.04805

Di Stefano, G., Gambardella, A., & Verona, G. (2012). Technology push and demand pull perspectives in innovation studies: Current findings and future research directions. *Research Policy*, *41*(8), 1283–1295. https://doi.org/https://doi.org/10.1016/j.respol.2012.03.021

Easterby-Smith, M., & Prieto, I. M. (2008, September). Dynamic capabilities and knowledge management: An integrative role for learning? https://doi.org/10.1111/j.1467-8551.2007.00543.x

Edge, D., Trinh, H., Cheng, N., Bradley, J., Chao, A., Mody, A., Truitt, S., Metropolitansky, D., Ness, R. O., & Larson, J. (2025). From local to global: A graph rag approach to query-focused summarization. https://arxiv.org/abs/2404.16130

Eisenhardt, K. M., & Martin, J. A. (2000). Dynamic capabilities: What are they? *Strategic Management Journal*, *21*(10-11), 1105–1121. https://doi.org/https://doi.org/10.1002/1097-0266(200010/11)21:10/11<1105::AID-SMJ133>3.0.CO;2-E

Ettinger, A. (2025). Enterprise architecture as a dynamic capability for scalable and sustainable generative ai adoption: Bridging innovation and governance in large organisations. https://arxiv.org/abs/2505.06326

European Parliament and of the Council. (2016). *General Data Protection Regulation*. https://eur-lex.europa.eu/legal-content/EN/ALL/?uri=celex:32016R0679

Fassnacht, M., Benz, C., Heinz, D., Leimstoll, J., & Satzger, G. (2023). Barriers to data sharing among private sector organizations. https://doi.org/10.24251/HICSS.2023.453

Fernsel, L., Kalff, Y., & Simbeck, K. (2024). Assessing the auditability of ai-integrating systems: A framework and learning analytics case study. https://arxiv.org/abs/2411.08906

Gao, L., Ma, X., Lin, J., & Callan, J. (2022). Precise zero-shot dense retrieval without relevance labels. http://arxiv.org/abs/2212.10496

Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., Dai, Y., Sun, J., Wang, M., & Wang, H. (2024). Retrieval-augmented generation for large language models: A survey. http://arxiv.org/abs/2312.10997

Gao, Y., Xiong, Y., Zhong, Y., Bi, Y., Xue, M., & Wang, H. (2025). Synergizing rag and reasoning: A systematic review. https://arxiv.org/abs/2504.15909

Gartner. (2025). Gartner forecasts worldwide genai spending to reach 644*Billionin*2025 [Retrieved 20 December, 2025]. https://www.gartner.com/en/newsroom/press-releases/2025-03-31-gartner-forecasts-worldwide-genai-spending-to-reach-644-billion-in-2025

Goodhue, D. L., & Thompson, R. L. (1995). Task-technology fit and individual performance. *MIS Quarterly*, *19*(2), 213–236. Retrieved November 7, 2025, from http://www.jstor.org/stable/249689

Grant, R. M. (1991). The resource-based theory of competitive advantage: Implications for strategy formulation. *California Management Review*, *33*(3), 114–135. https://doi.org/10.2307/41166664

Grant, R. M. (1996). Toward a knowledge-based theory of the firm. *Strategic Management Journal*, *17*, 109–122. https://doi.org/10.1002/smj.4250171110

Guba, E. G., & Lincoln, Y. S. (1985). *Naturalistic inquiry*. Sage Publications.

Guerraoui, R., Kermarrec, A.-M., Petrescu, D., Pires, R., Randl, M., & de Vos, M. (2025). Efficient federated search for retrieval-augmented generation. https://arxiv.org/abs/2502.19280

Guo, Y., Bian, H., Zhou, L., Wang, Z., Zhang, Z., Kawala, F., Dean, M., Fischer, I., Peng, Y., Tokgozoglu, N., Barrientos, I., Shaik, R., Li, R., Venkataraman, C., Far, R. S., Pawar, M., Sundaranatha, V.,

Xu, M., & Chu, F. (2025). Adversarial distilled retrieval-augmented guarding model for online malicious intent detection. http://arxiv.org/abs/2509.14622

Halawi, L. A., Aronson, J. E., & McCarthy, R. V. (2005). Resource-based view of knowledge management for competitive advantage. https://academic-publishing.org/index.php/ejkm/article/view/724

Haldin-Herrgård, T. (2000). Difficulties in diffusion of tacit knowledge in organizations. *Journal of Intellectual Capital*, *1*, 357–365. https://doi.org/10.1108/14691930010359252

Han, H., Wang, Y., Shomer, H., Guo, K., Ding, J., Lei, Y., Halappanavar, M., Rossi, R. A., Mukherjee, S., Tang, X., He, Q., Hua, Z., Long, B., Zhao, T., Shah, N., Javari, A., Xia, Y., & Tang, J. (2025). Retrieval-augmented generation with graphs (graphrag). http://arxiv.org/abs/2501.00309

Hasan, M. T., Waseem, M., Kemell, K.-K., Khan, A. A., Saari, M., & Abrahamsson, P. (2025, September). Engineering rag systems for real-world applications: Design, development, and evaluation. In *Software engineering and advanced applications* (pp. 143–158). Springer Nature Switzerland. https://doi.org/10.1007/978-3-032-04200-2_10

Henderson, J., & Venkatraman, N. (1994). Strategic alignment: A model for organizational transformation via information technology. *Information Technology and the Corporation of the 1990s Research Studies*. https://doi.org/10.1093/oso/9780195068061.003.0009

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in information systems research. *MIS Q.*, *28*(1), 75–105.

Idrees, H., Xu, J., Haider, S. A., & Tehseen, S. (2023). A systematic review of knowledge management and new product development projects: Trends, issues, and challenges. *Journal of Innovation and Knowledge*, *8*. https://doi.org/10.1016/j.jik.2023.100350

Jin, Z., Cao, P., Chen, Y., Liu, K., Jiang, X., Xu, J., Li, Q., & Zhao, J. (2024). Tug-of-war between knowledge: Exploring and resolving knowledge conflicts in retrieval-augmented language models. http://arxiv.org/abs/2402.14409

Katsamakas, E., & Xin, M. (2019). Open source adoption strategy. *Electronic Commerce Research and Applications*, *36*, 100872. https://doi.org/https://doi.org/10.1016/j.elerap.2019.100872

Kim, J. S., Kim, M., & Baek, T. H. (2025). Enhancing user experience with a generative ai chatbot. *International Journal of Human–Computer Interaction*, *41*(1), 651–663. https://doi.org/10.1080/10447318.2024.2311971

Koivisto, K., & Taipalus, T. (2023). Pitfalls in effective knowledge management: Insights from an international information technology organization. https://doi.org/10.48550/arXiv.2304.07737

Kotha, R. (2022). Hybrid cloud solutions for balancing on-premise and cloud infrastructure. *Journal of Mathematical Computer Applications*, 1–5. https://doi.org/10.47363/JMCA/2022(1)E109

Kraaijenbrink, J., Spender, J.-C., & Groen, A. J. (2010). The resource-based view: A review and assessment of its critiques. *Journal of Management*, *36*(1), 349–372. https://doi.org/10.1177/0149206309350775

Krishna, S., Krishna, K., Mohananey, A., Schwarcz, S., Stambler, A., Upadhyay, S., & Faruqui, M. (2025). Fact, fetch, and reason: A unified evaluation of retrieval-augmented generation. http://arxiv.org/abs/2409.12941

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., Riedel, S., & Kiela, D. (2021). Retrieval-augmented generation for knowledge-intensive nlp tasks. http://arxiv.org/abs/2005.11401

Li, D., Yu, G., Wang, X., & Liang, B. (2025). Auditablellm: A hash-chain-backed, compliance-aware auditable framework for large language models. *Electronics*. https://doi.org/10.3390/electronics15010056

Liu, N. F., Lin, K., Hewitt, J., Paranjape, A., Bevilacqua, M., Petroni, F., & Liang, P. (2023). Lost in the middle: How language models use long contexts. http://arxiv.org/abs/2307.03172

Lupinacci, M., Pironti, F. A., Blefari, F., Romeo, F., Arena, L., & Furfaro, A. (2025). The dark side of llms: Agent-based attacks for complete computer takeover. https://arxiv.org/abs/2507.06850

McKinsey & Company. (2024). *What is retrieval-augmented generation (rag)?* (Tech. rep.) (Retrieved 13 January, 2026). https://www.mckinsey.com/featured-insights/mckinsey-explainers/what-is-retrieval-augmented-generation-rag?

McKinsey & Company. (2025, September). *Beyond the hype: Unlocking value from the ai revolution* (tech. rep.) (Retrieved 13 January, 2026). https://www.mckinsey.com/cn/our-insights/our-insights/beyond-the-hype-unlocking-value-from-the-ai-revolution

Menlo Ventures. (2024, November). *2024: The state of generative ai in the enterprise* [Retrieved 18 August, 2025]. https://menlovc.com/2024-the-state-of-generative-ai-in-the-enterprise/

Mickie, J., & Elson, A. (2025). Optimizing ai workflows: Real-time data processing and machine learning for scalable systems. https://doi.org/10.13140/RG.2.2.28647.15526

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. http://arxiv.org/abs/1301.3781

Mohammad, Y., Nachouki, M., & Mohamed, E. A. (2024). Knowledge management systems in business management using knowledge graphs and semantic technologies. *International Journal of Knowledge Management*, *21*(1). https://doi.org/https://doi.org/10.4018/IJKM.369121

Mohammadabadi, S. M., Kara, B. C., Eyupoglu, C., Uzay, C., Tosun, M. S., & Karakuş, O. (2025). A survey of large language models: Evolution, architectures, adaptation, benchmarking, applications, challenges, and societal implications. *Electronics*, *14*(18). https://doi.org/10.3390/electronics14183580

Müller, L., Holstein, J., Bause, S., Satzger, G., & Kühl, N. (2025). Data quality challenges in retrieval-augmented generation. https://arxiv.org/abs/2510.00552

Nakash, M., & Bolisani, E. (2024). Knowledge management meets artificial intelligence: A systematic review and future research agenda. *European Conference on Knowledge Management*, *25*, 544–552. https://doi.org/10.34190/eckm.25.1.2443

Narajala, V. S., & Narayan, O. (2025). Securing agentic ai: A comprehensive threat model and mitigation framework for generative ai agents. https://arxiv.org/abs/2504.19956

Nielsen, B., & Michailova, S. (2007). Knowledge management systems in multinational corporations: Typology and transitional dynamics. *Long Range Planning*, *40*, 314–340. https://doi.org/10.1016/j.lrp.2007.04.005

Nonaka, I., & Takeuchi, H. (1995, May). *The knowledge-creating company: How japanese companies create the dynamics of innovation*. Oxford University Press. https://doi.org/10.1093/oso/9780195092691.001.0001

Oche, A. J., Folashade, A. G., Ghosal, T., & Biswas, A. (2025). A systematic review of key retrieval-augmented generation (rag) systems: Progress, gaps, and future directions. http://arxiv.org/abs/2507.18910

OECD. (2025). *Governing with artificial intelligence: The state of play and way forward in core government functions* (tech. rep.). OECD Publishing. https://doi.org/10.1787/795de142-en

OECD. (2026). Enterprises by business size [Retrieved on 16 January, 2026]. https://www.oecd.org/en/data/indicators/enterprises-by-business-size.html

Oliveira, T., Martins, R., Sarker, S., Thomas, M., & Popovič, A. (2019). Understanding saas adoption: The moderating impact of the environment context. *International Journal of Information Management*, *49*, 1–12. https://doi.org/https://doi.org/10.1016/j.ijinfomgt.2019.02.009

Opara-Martins, J., Sahandi, R., & Tian, F. (2016). Critical analysis of vendor lock-in and its impact on cloud computing migration: A business perspective. *Journal of Cloud Computing*, *5*. https://doi.org/10.1186/s13677-016-0054-z

Ouyang, J., Pan, T., Cheng, M., Yan, R., Luo, Y., Lin, J., & Liu, Q. (2025). Hoh: A dynamic benchmark for evaluating the impact of outdated information on retrieval-augmented generation. https://arxiv.org/abs/2503.04800

Packowski, S., Halilovic, I., Schlotfeldt, J., & Smith, T. (2024). Optimizing and evaluating enterprise retrieval-augmented generation (rag): A content design perspective. http://arxiv.org/abs/2410.12812

Pan, G., & Wang, H. (2025). A cost-benefit analysis of on-premise large language model deployment: Breaking even with commercial llm services. http://arxiv.org/abs/2509.18101

Peng, B., Zhu, Y., Zhang, Y., Bo, X., Liu, Y., Shi, H., Hong, C., & Tang, S. (2024). Graph retrieval-augmented generation: A survey. https://doi.org/10.48550/arXiv.2408.08921

Petrolini, M., Cagnoni, S., & Mordonini, M. (2022). Automatic detection of sensitive data using transformer-based classifiers. *Future Internet*, *14*. https://doi.org/10.3390/fi14080228

Poliakov, M., & Shvai, N. (2024). Multi-meta-rag: Improving rag for multi-hop queries using database filtering with llm-extracted metadata. https://doi.org/10.1007/978-3-031-81372-6_25

Porter, M. E. (1979). How competitive forces shape strategy. *Harvard Business Review*, 137–145.

Porter, M. E. (1980). *Competitive strategy: Techniques for analyzing industries and competitors*. Free Press.

RAGFlow. (2025, May). Rag at the crossroads: Mid-2025 reflections on ai evolution [Retrieved 10 December, 2025]. https://ragflow.io/blog/rag-at-the-crossroads-mid-2025-reflections-on-ai-evolution

Ren, Z., Doekemeijer, K., Apparao, P., & Trivedi, A. (2025). Storage-Based Approximate Nearest Neighbor Search: What are the Performance, Cost, and I/O Characteristics? *2025 IEEE International Symposium on Workload Characterization (IISWC)*, 407–422. https://doi.org/10.1109/IISWC66894.2025.00041

Robertson, J., Caruana, A., & Ferreira, C. (2023). Innovation performance: The effect of knowledge-based dynamic capabilities in cross-country innovation ecosystems. *International Business Review*, *32*. https://doi.org/10.1016/j.ibusrev.2021.101866

Roychowdhury, S., Soman, S., Gireesha, R. H., Chhabra, V., Gunda, N., Bandyopadhyay, S., & Bala, S. K. (2025). Investigating distributions of telecom adapted sentence embeddings for document retrieval. https://arxiv.org/abs/2406.12336

Shen, M., Umar, M., Maeng, K., Suh, G. E., & Gupta, U. (2024). Towards understanding systems trade-offs in retrieval-augmented generation model inference. http://arxiv.org/abs/2412.11854

Singh, A., Ehtesham, A., Kumar, S., & Khoei, T. T. (2025). Agentic retrieval-augmented generation: A survey on agentic rag. http://arxiv.org/abs/2501.09136

Soomro, R. B., Al-Rahmi, W. M., Dahri, N. A., Almuqren, L., Al-mogren, A. S., & Aldaijy, A. (2025). A sem–ann analysis to examine impact of artificial intelligence technologies on sustainable performance of smes. *Scientific Reports*, *15*(1), 5438. https://doi.org/10.1038/s41598-025-86464-3

Spender, J.-C. (1996). Making knowledge the basis of a dynamic theory of the firm. *Strategic Management Journal*, *17*(S2), 45–62. https://doi.org/10.1002/smj.4250171106

Tan, Z., Zhao, C., Moraffah, R., Li, Y., Wang, S., Li, J., Chen, T., & Liu, H. (2024). "glue pizza and eat rocks" – exploiting vulnerabilities in retrieval-augmented generative models. https://arxiv.org/abs/2406.19417

Teece, D. J. (2007). Explicating dynamic capabilities: The nature and microfoundations of (sustainable) enterprise performance. *Strategic Management Journal*, *28*, 1319–1350. https://doi.org/10.1002/smj.640

Teece, D. J., Pisano, G., & Shuen, A. (1997). Dynamic capabilities and strategic management. *Strategic Management Journal*, *18*(7), 509–533. https://doi.org/https://doi.org/10.1002/(SICI)1097-0266(199708)18:7<509::AID-SMJ882>3.0.CO;2-Z

Tikkinen-Piri, C., Rohunen, A., & Markkula, J. (2018). Eu general data protection regulation: Changes and implications for personal data collecting companies. *Computer Law & Security Review*, *34*, 134–153. https://doi.org/https://doi.org/10.1016/j.clsr.2017.05.015

Tornatzky, L. G., Fleischer, M., & Chakrabarti, A. K. (1990). *Processes of technological innovation*. Lexington Books.

Torraco, R. J. (2005). Writing integrative literature reviews: Guidelines and examples. *Human Resource Development Review*, *4*(3), 356–367. https://doi.org/10.1177/1534484305278283

Ulčar, M., Žagar, A., Armendariz, C. S., Repar, A., Pollak, S., Purver, M., & Robnik-Šikonja, M. (2026). Mono- and cross-lingual evaluation of representation language models on less-resourced languages. *Computer Speech and Language*, *95*. https://doi.org/10.1016/j.csl.2025.101852

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, & R. Garnett, Eds.). *30*. https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

Wang, C. L., & Ahmed, P. K. (2007). Dynamic capabilities: A review and research agenda. *International Journal of Management Reviews*, *9*(1), 31–51. https://doi.org/https://doi.org/10.1111/j.1468-2370.2007.00201.x

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., & Zhou, D. (2023). Chain-of-thought prompting elicits reasoning in large language models. http://arxiv.org/abs/2201.11903

Wernerfelt, B. (1984). A resource-based view of the firm. *Strategic Management Journal*, *5*(2), 171–180. Retrieved September 10, 2025, from http://www.jstor.org/stable/2486175

Wilden, R., Devinney, T. M., & Dowling, G. R. (2016). The architecture of dynamic capability research identifying the building blocks of a configurational approach. *The Academy of Management Annals*, *10*(1), 997–1076. https://doi.org/10.1080/19416520.2016.1161966

Xiao, Y., Guo, J., Fan, Y., Lan, Y., Xu, J., & Cheng, X. (2018). Beyond precision: A study on recall of initial retrieval with neural representations. https://arxiv.org/abs/1806.10869

Xu, X., Weytjens, H., Zhang, D., Lu, Q., Weber, I., & Zhu, L. (2025). Ragops: Operating and managing retrieval-augmented generation pipelines. http://arxiv.org/abs/2506.03401

Yang, J., Blount, Y., & Amrollahi, A. (2024). Artificial intelligence adoption in a professional service industry: A multiple case study. *Technological Forecasting and Social Change*, *201*, 123251. https://doi.org/https://doi.org/10.1016/j.techfore.2024.123251

Yuan, L., Han, D.-J., Wang, S., & Brinton, C. G. (2025). Local-cloud inference offloading for llms in multi-modal, multi-task, multi-dialogue settings. http://arxiv.org/abs/2502.11007

Zhang, X., Pang, Y., Kang, Y., Chen, W., Fan, L., Jin, H., & Yang, Q. (2025). No free lunch theorem for privacy-preserving llm inference. *Artificial Intelligence*, *341*, 104293. https://doi.org/https://doi.org/10.1016/j.artint.2025.104293

Zhao, S., Yang, Y., Wang, Z., He, Z., Qiu, L. K., & Qiu, L. (2024). Retrieval augmented generation (rag) and beyond: A comprehensive survey on how to make your llms use external data more wisely. http://arxiv.org/abs/2409.14924

Zhao, Y., Wen, S., Zhou, T., Liu, W., Yu, H., & Xu, H. (2022). Development and innovation of enterprise knowledge management strategies using big data neural networks technology. *Journal of Innovation Knowledge*, *7*(4), 100273. https://doi.org/https://doi.org/10.1016/j.jik.2022.100273

Zhou, T., & Lu, H. (2024). The effect of trust on user adoption of ai-generated content. *The Electronic Library*, *43*. https://doi.org/10.1108/EL-08-2024-0244

Zhu, Y., Yuan, H., Wang, S., Liu, J., Liu, W., Deng, C., Chen, H., Liu, Z., Dou, Z., & Wen, J.-R. (2024). Large language models for information retrieval: A survey. http://arxiv.org/abs/2308.07107

Zirar, A., Ali, S. I., & Islam, N. (2023). Worker and workplace artificial intelligence (ai) coexistence: Emerging themes and research agenda. *Technovation*, *124*. https://doi.org/10.1016/j.technovation.2023.102747

Zollo, M., & Winter, S. G. (2002). Deliberate learning and the evolution of dynamic capabilities. *Organization Science*, *13*, 339–351. https://doi.org/10.1287/orsc.13.3.339.2780

Appendix A

Taxonomy of Foundational RAG Decisions.

Table A.1: Taxonomy of foundational RAG design decisions

Category	Description	Key Trade-offs
Strategic Decisions		
Query Complexity Level	Maximum query level the system must handle (Level 1-4)	Performance vs capability requirements
KBDC Capabilities	Capabilities the system must enable	Scope vs resource requirements
Architecture Decisions		
RAG Paradigm	Naive/Advanced/Agentic approach	Capability vs complexity vs cost
Embedding Model	Pretrained, fine-tuned, or domain-specific	Retrieval accuracy vs cost vs development time
User Interface Model	Chat with context vs search-style single queries	User experience vs architectural complexity
Knowledge Base Decisions		
Knowledge Base Scope	Which documents and sources to include	Coverage vs retrieval precision vs governance overhead
External Data Access	Including access to knowledge that lies outside the organisation	Coverage vs complexity vs security
Multimodal Support	Text-only or include image, video, code, audio, etc.	Coverage vs ingestion complexity vs cost
Multilingual Support	Monolingual vs multilingual vs translation	Performance vs coverage vs complexity
Knowledge Base Freshness	Real-time, batch, or manual refresh	Freshness vs stability vs cost
Infrastructure Decisions		
Deployment Model	On-premises/Cloud/Hybrid	Control vs scalability vs cost
Technology Stack	Build vs buy vs open-source vs proprietary	Customisation vs time to market vs cost
Retrieval Architecture	Vector-based, graph-based or hybrid	Performance vs cost vs control vs features
Model Hosting	Commercial APIs or self-host	Convenience vs control vs cost profile
Governance Decisions		

Continued on next page

Table A.1 – continued from previous page

Category	**Description**	**Key Trade-offs**
Access Control	Permission enforcement in RAG	Security vs usability vs governance overhead
Personal Information	Handling of personal information	Capability vs compliance complexity
Regulatory Compliance	Regulations such as GDPR or domain-specific requirements	Capability vs overhead
Security Posture	Defence coverage and mechanisms	Safety vs complexity vs latency
Audit Trail Scope	What system activities to log and retain	Accountability vs storage cost vs governance

Appendix B

Key Findings per Framework Dimension

Table B.1: Key empirical findings per framework dimension.

Framework element	Key Finding From Interviews
Strategic Considerations	
Productivity Improvement	Primary strategic motivation across organisations. They seek faster information access and reduced time spent on manual document search
Knowledge Discovery	Strategic goal to reduce organisational silos, revealing knowledge across previously isolated domains
Creating AI Capability	Strategic positioning for future competitive advantage through internal capability development and learning-by-doing. Often done through a technology push with no clear use cases present
Foundational Design Decisions	
Domain vs General Scope	Strong convergence on domain-specific approaches, as practitioners argue these create more value
Buy vs Build	Strong practitioner preference for building custom systems due to governance control needs, limited vendor capabilities, and desire to develop internal expertise
Access Control Design	Determines whether systems can scale beyond pilot stages; robust access control enables scaling across teams and domains
Infrastructure Architecture	Can be predetermined by existing legacy systems, IT policies, and regulatory requirements rather than actively chosen during RAG implementation
RAG Architecture Complexity	Practitioners strongly favour simple vector-based systems over sophisticated agentic approaches, arguing that complex architectures are unnecessary when simple systems achieve satisfactory performance
Success Criteria	Defined success metrics and a strong business case are needed for RAG success
Operational Design Decisions	
LLM Choice	Relatively reversible decision involving cost-latency-reasoning trade-offs
Chunk Strategy	Simple chunking strategies often suffice for enterprise needs; over-optimisation reduces scalability
Prompt Engineering	Simple prompts achieve adequate performance; extensive edge-case handling creates maintenance burdens and reduces scalability
Retrieval Strategy	Simple hybrid approaches combining keyword and semantic search typically deliver satisfactory results
Interface Design	GenAI systems require strong UX/UI, conversational interfaces demand design attention that traditional ML systems do not
Adoption Enablers	

Continued on next page

Table B.1 – continued from previous page

Framework Element	Key Finding From Interviews
Communication	Clear messaging about system capabilities and limitations prevent expectation mismatches that undermine user trust and adoption
Earning Trust	Occasional hallucinations will damage adoption in enterprise contexts, but cannot be fully mitigated
AI Literacy	Users require education on effective prompting, interpreting responses, recognising system limitations, and verifying answers
Ease of Use	Latency, interface quality, and workflow integration are critical for RAG adoption.
Value Delivery	Measurable outcomes justify behavioural change from manual search to RAG powered retrieval
Organisational Constraints	
Data Quality & Ownership	Forms the primary constraint for RAG success. Poor formatting, staleness, and unclear ownership undermine RAG regardless of architectural sophistication
Governance & Access Control	Internal access control policy maturity determines whether RAG can safely scale beyond limited domains
Stakeholder Management	Requires coordination across technical teams, business units, risk departments, and management, each with distinct priorities
Technological Constraints	
Evaluation Challenges	It is difficult to measure RAG quality beyond retrieval accuracy; automated metrics do not fully capture answer usefulness. Organisations rely on adoption metrics and human feedback as proxies for system quality
IT Capabilities	Knowledge of machine learning, GenAI, and AI operations are required for custom development, but represent the least binding constraint for organisations with mature IT departments
Environmental Constraints	
Compliance & Regulation	External regulatory requirements (GDPR, data residency laws, industry-specific regulations) constrain infrastructure choices and data handling practices

Appendix C

Interview Guide

Context & Background

1. What is your role, and how were you involved with this RAG solution?
2. What was the original business case or problem you were trying to solve?
3. What maturity stage is the system at now?
4. How long has development been ongoing?
5. Who are the intended users, and what should they accomplish with the system?

Strategic Intent & Scope

6. How did you decide on the initial scope of the system?
7. Has the scope changed during development, and why?
8. What types of questions did you design the system to answer?
9. Are you building for a single domain or trying to scale across multiple domains?
10. What's your vision for where this system should be in 1-2 years?

Knowledge Base Decisions

11. How did you decide which data sources to include in the knowledge base?
12. What data quality challenges did you encounter?
13. Who owns and maintains the data that goes into the system?
14. How do you keep the knowledge base fresh and up-to-date?

Architecture & Infrastructure

15. Walk me through your current RAG architecture - is it naive, advanced, or agentic?
16. How did you decide on build vs. buy vs. using existing platforms?
17. What deployment model did you choose and why?
18. Did you consider using something like Copilot, or was custom development always the plan?

Governance & Security

19. How did you handle access control and permissions?
20. What governance challenges did you run into, if any?
21. How did compliance or regulatory requirements influence your design?

Challenges & Lessons Learned

22. What was the biggest challenge you faced during development?
23. What decisions made early on needed to be changed later?
24. What's the most common mistake you see organisations make with RAG?
25. If you were starting over, what would you do differently?
26. What organisational factors made implementation easier or harder?

Adoption & Performance

27. How has user adoption been so far?
28. How do you measure whether the system is successful?
29. What feedback have you gotten from users?

Future & Scaling

30. How do you see RAG evolving in the coming years?
31. What are your plans for scaling - more users in this domain, expanding to other domains, or integrating with agents?

Appendix D

Code Distribution Heatmaps

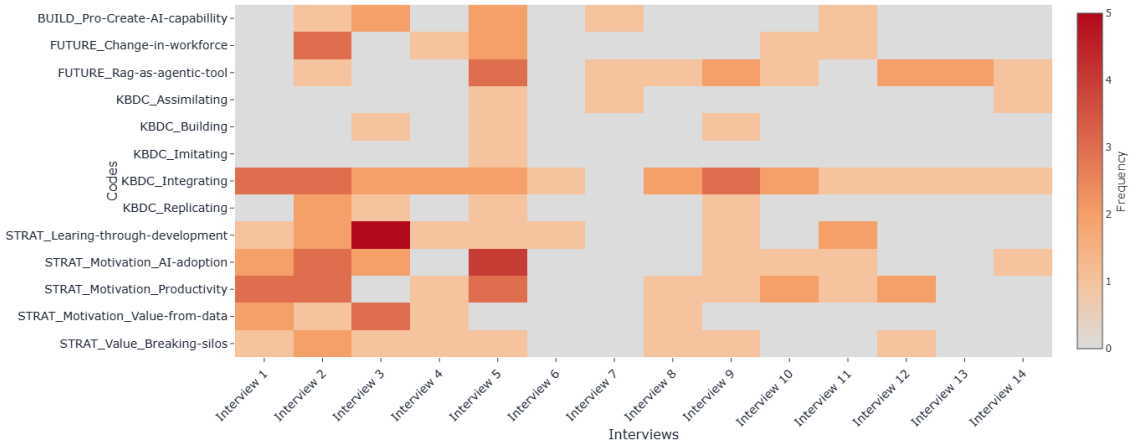

Figure D.1: A heatmap showing the code frequencies for theme 1.

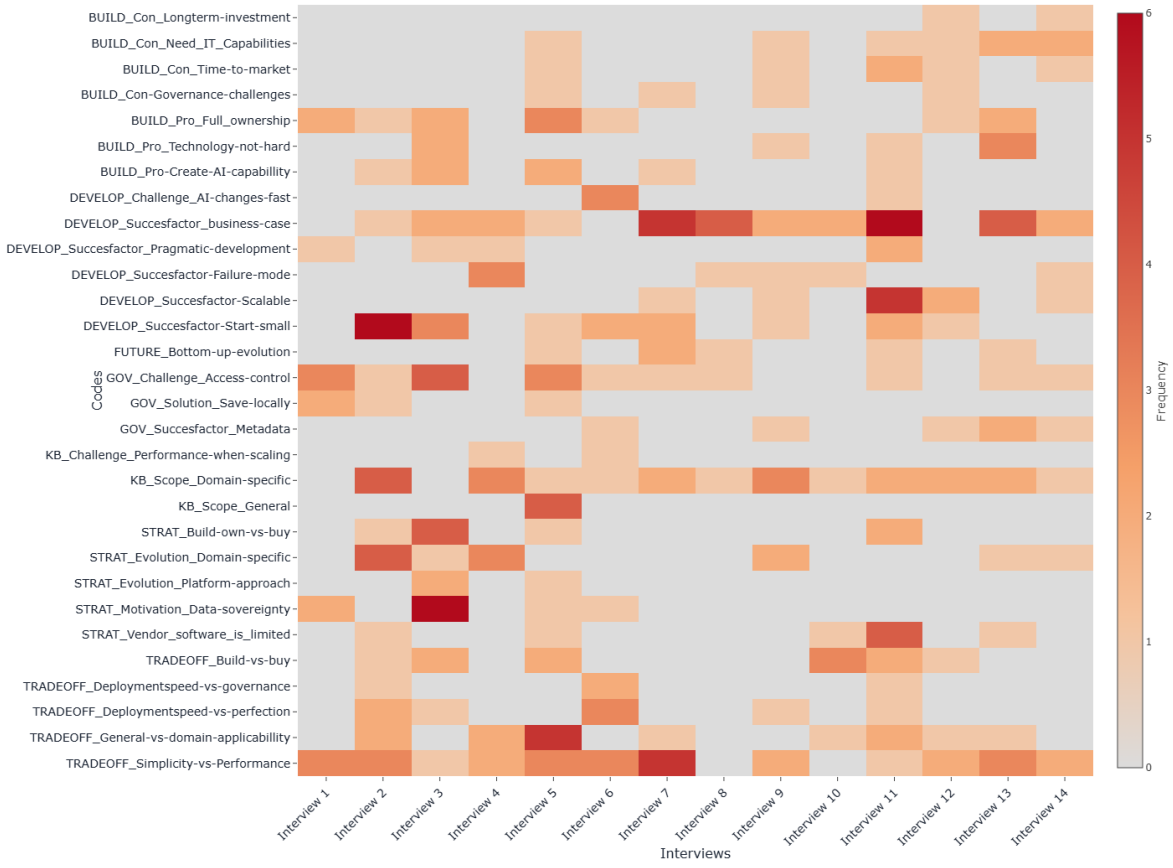

Figure D.2: A heatmap showing the code frequencies for theme 2.

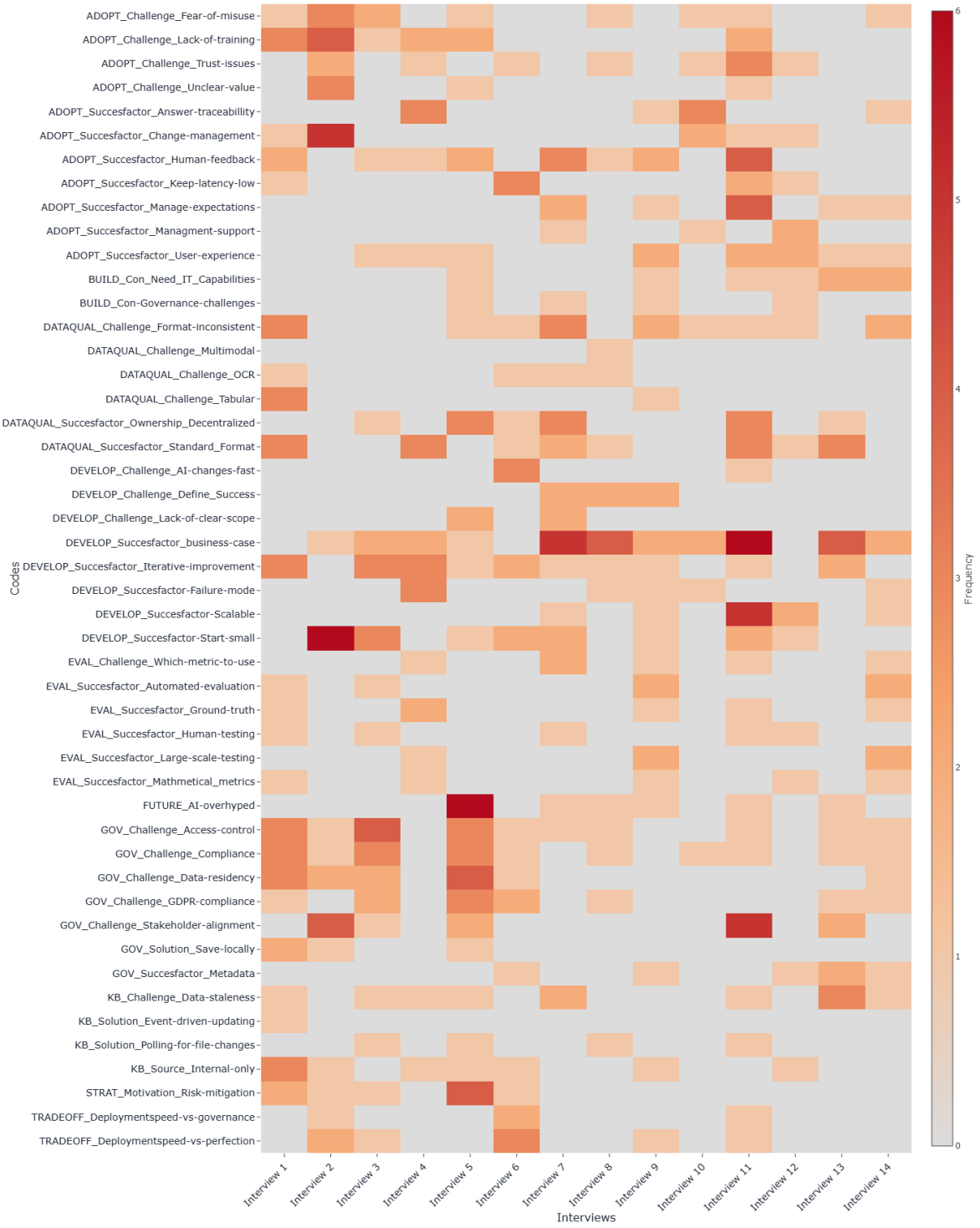

Figure D.3: A heatmap showing the code frequencies for theme 3.

Appendix E

Codebook

Table E.1: The complete codebook that resulted from the thematic analysis.

Code	Definition
ADOPT – User Adoption (11 codes)	
ADOPT_Challenge_Fear-of-misuse	Employees afraid of misusing RAG and GenAI
ADOPT_Challenge_Lack-of-training	Don't know how to use effectively or what it can do
ADOPT_Challenge_Trust-issues	Users don't trust AI output accuracy
ADOPT_Challenge_Unclear-value	Cannot see practical benefit for their work
ADOPT_Successfactor_Answer-traceability	Source traceability crucial for compliance and trust
ADOPT_Successfactor_Change-management	Employees need to change their behaviour and embed AI in their work
ADOPT_Successfactor_Human-feedback	Improving the RAG answers and user experience by gathering human feedback
ADOPT_Successfactor_Keep-latency-low	Systems need to maintain low latency to ensure user satisfaction
ADOPT_Successfactor_Manage-expectations	Developers need to manage user expectations and communicate what the system can and cannot do
ADOPT_Successfactor_Management-support	RAG systems need the support from management to drive adoption
ADOPT_Successfactor_User-experience	User experience is essential to ensure RAG adoption
BUILD – Build vs Buy Decision (7 codes)	
BUILD_Con_Longterm-investment	Building custom RAG requires both development and operational costs
BUILD_Con_Need_IT_Capabilities	To build a custom RAG, you need proficient IT, ML and AI capabilities
BUILD_Con_Time-to-market	Time-to-market with custom development takes longer than buying an off-the-shelf solution
BUILD_Con-Governance-challenges	Building custom RAG solutions introduces governance challenges such as access control and compliance
BUILD_Pro_Full_Ownership	Custom RAGs allow full control over the created solution, which is not possible with third-party RAG
BUILD_Pro_Technology-not-hard	RAG is relatively simple to build in terms of technology, especially a proof of concept

Continued on next page

Table E.1 – *Continued from previous page*

Code	Definition
BUILD_Pro-Create-AI-capability	Building custom RAG is an 'easy' introduction to GenAI in an organisation
DATAQUAL – Data Quality (6 codes)	
DATAQUAL_Challenge_Format-inconsistent	Inconsistent document formatting across sources
DATAQUAL_Challenge_Multimodal	Handling images, charts, graphs within documents
DATAQUAL_Challenge_OCR	Historical or scanned documents requiring OCR processing
DATAQUAL_Challenge_Tabular	Difficulty processing tables and structured data effectively
DATAQUAL_Successfactor_Ownership_Decentralized	Data owners in each department maintain their own quality
DATAQUAL_Successfactor_Standard_Format	Data needs to be in a specific readable format
DEVELOP – Development Process (9 codes)	
DEVELOP_Challenge_AI-changes-fast	AI changes fast; self-developed tools may become outdated quickly
DEVELOP_Challenge_Define_Success	It is hard for organisations to define when a RAG setup is actually successful, especially in advance
DEVELOP_Challenge_Lack-of-clear-scope	Lots of stakeholders are involved and want their own data added; scope creep can become a problem
DEVELOP_Successfactor_business-case	Creating a strong business case and understanding the problem that needs to be solved leads to success
DEVELOP_Successfactor_Iterative-improvement	Iteratively improving the RAG pipeline and the layers around it
DEVELOP_Successfactor_Pragmatic-development	Taking a practical, pragmatic approach to development rather than over-engineering
DEVELOP_Successfactor-Failure-mode	Models should be clear in their scope and be transparent if the answer is not in their knowledge base
DEVELOP_Successfactor-Scalable	The system needs to be scalable beyond the original business case
DEVELOP_Successfactor-Start-small	Starting small with a limited scope enables quick wins; you can always expand later
EVAL – Evaluation (6 codes)	
EVAL_Challenge_Which-metric-to-use	There can be many metrics for evaluation, but which ones actually tell us the most?
EVAL_Successfactor_Automated-evaluation	Automating the evaluation allows continuous monitoring
EVAL_Successfactor_Ground-truth	Having ground truth data to evaluate your RAG pipeline
EVAL_Successfactor_Human-testing	Allow users to test the system and give feedback
EVAL_Successfactor_Large-scale-testing	Don't look at the performance of 5 questions, but of 10,000. Some questions will always be answered incorrectly. Don't need 100%, 90% is good
EVAL_Successfactor_Mathematical_metrics	Using metrics such as faithfulness, correctness, BLEU, etc.
FUTURE – Future Outlook (4 codes)	
FUTURE_AI-overhyped	Practitioners feel AI is overhyped, but has value in specific cases

Continued on next page

Table E.1 – *Continued from previous page*

Code	Definition
FUTURE_Bottom-up-evolution	Best way for RAG to evolve is to let it evolve bottom-up; domains build own solutions, and will maybe be merged later in the future
FUTURE_Change-in-workforce	RAG and AI will change workforce dynamics and job requirements
FUTURE_Rag-as-agentic-tool	RAG will become a tool for agents instead of standalone applications
GOV – Governance (7 codes)	
GOV_Challenge_Access-control	Managing who can access which data is the primary difficulty
GOV_Challenge_Compliance	Privacy and compliance are the dominant constraints, not technical
GOV_Challenge_Data-residency	Constraints on where data can be physically stored
GOV_Challenge_GDPR-compliance	Meeting GDPR requirements for data handling
GOV_Challenge_Stakeholder-alignment	Multiple parties with different priorities and requirements
GOV_Solution_Save-locally	All processing and storage on local devices (no central storage)
GOV_Successfactor_Metadata	Using metadata tags to restrict query scope by permissions
KB – Knowledge Base (7 codes)	
KB_Challenge_Data-staleness	Information becomes outdated quickly without maintenance
KB_Challenge_Performance-when-scaling	KB performance drops as it grows in size
KB_Scope_Domain-specific	Focused knowledge base for specific practice or function
KB_Scope_General	Broad organisational knowledge base covering multiple domains
KB_Solution_Event-driven-updating	Updates triggered by document modification events
KB_Solution_Polling-for-file-changes	Automated checking for document changes (checksums, file size)
KB_Source_Internal-only	Knowledge base contains only internal organisational data
KBDC – Knowledge-Based Dynamic Capabilities (5 codes)	
KBDC_Assimilating	Searching, acquiring, and internalising external knowledge
KBDC_Building	Creating new knowledge by recombining existing internal knowledge
KBDC_Imitating	Adopting and adapting external knowledge or practices
KBDC_Integrating	Recognising, transferring, and integrating internal knowledge sources
KBDC_Replicating	Identifying and applying existing knowledge in other organisational parts
STRAT – Strategy (11 codes)	
STRAT_Build-own-vs-buy	Strategic decision to build custom RAG vs buying vendor solution

Continued on next page

Table E.1 – *Continued from previous page*

Code	Definition
STRAT_Evolution_Domain-specific	Strategic shift from general-purpose to domain-focused RAG
STRAT_Evolution_Platform-approach	Building infrastructure for multiple RAG applications vs single solution
STRAT_Learning-through-development	Using RAG development as a learning opportunity for AI capabilities
STRAT_Motivation_AI-adoption	Using RAG to accelerate organisational AI learning and build capability
STRAT_Motivation_Data-sovereignty	Maintaining control over organisational data and preventing external use
STRAT_Motivation_Productivity	Improving employee efficiency through AI-assisted knowledge work
STRAT_Motivation_Risk-mitigation	Adopting RAG to avoid risks from insecure external tools or privacy concerns
STRAT_Motivation_Value-from-data	Extracting more strategic value from existing organisational data
STRAT_Value_Breaking-silos	Connecting information across departments
STRAT_Vendor_software_is_limited	Recognition that vendor solutions lack customisation flexibility
TRADEOFF – Trade-offs (5 codes)	
TRADEOFF_Build-vs-buy	Custom development vs vendor solutions
TRADEOFF_Deploymentspeed-vs-governance	Tension between rapid deployment and full approvals and controls
TRADEOFF_Deploymentspeed-vs-perfection	Quick deployment vs comprehensive solution
TRADEOFF_General-vs-domain-applicability	Broad applicability vs optimised domain performance
TRADEOFF_Simplicity-vs-Performance	Simple maintainable architecture vs feature-rich complex systems