Enhancing Tool Orchestration and Planning for Agentic RAG Systems: A Case Study

Renato Miyaji[1], **Vitor Bedin**[1], **Renato Moulin**[1], **Leonardo Machado**[1]

[1]Grupo Visagio

{renato.miyaji,vitor.bedin,renato.moulin,leonardo.machado}@visagio.com

Abstract. *Traditional RAG faces bottlenecks in industrial environments. To address these limitations, this paper proposes an Agentic RAG architecture that decouples high-level strategic orchestration from technical execution through an LLM Supervisor and domain-specialized sub-agents. Using a proprietary dataset, we evaluate the efficacy of System 1 and System 2 reasoning paradigms alongside tool integration. Our experimental results demonstrate that ReWOO combined with Few-Shot execution guidance achieves a maximum success rate of 93.33%. Furthermore, we introduce an evaluation methodology which reveals that the ToolVerifier module is essential for complex planning. This study establishes that modular planning and robust verification are critical for reliability in knowledge-intensive industrial applications.*

1. Introduction

Although Large Language Models (LLMs) have revolutionized natural language processing, their reliance on pre-training data often results in hallucinations, a limitation mitigated by Retrieval-Augmented Generation (RAG). By employing strategies such as query augmentation and reranking, these systems ground model outputs in external knowledge; nevertheless, they encounter computational and accuracy bottlenecks when scaled to industrial environments characterized by vast volumes of multimodal data [Gao et al. 2025]. To address the inadequacies, the paradigm has shifted toward Agentic RAG, which embeds decision-making, dynamic search strategies, and adaptive tool use. This evolution facilitates iterative contextual refinement and multi-step reasoning.

Central to this paradigm is the necessary shift from basic reactive models to deliberative System 2 reasoning [Kahneman 2011], employing techniques like ReWOO [Xu et al. 2023], SwiftSage [Lin et al. 2023], and Reflexion [Shinn et al. 2023] to optimize planning trajectories and orchestrate tools in multi-hop queries. Although integrating specialized tools broadens agent capabilities, ensuring the semantic and syntactic integrity of these invocations requires robust verification mechanisms [Mekala et al. 2024]. Furthermore, because existing evaluation methods conflate cognitive planning quality with individual search tool performance, this research proposes a hierarchical multi-agent architecture designed to decouple high-level strategic orchestration from technical execution. By employing an LLM Supervisor alongside four domain-specialized sub-agents to navigate over multimodal documents from a proprietary consulting dataset, this study provides a real-world evaluation of efficacy for dynamic external tool integration and complex industrial information retrieval.

This work offers four primary contributions. First, it introduces a hierarchical framework that utilizes advanced planning paradigms to minimize computational overhead

and error propagation in multi-hop reasoning tasks. Second, it implements verification mechanisms to ensure the semantic and syntactic integrity of sub-agents and tool calls. Third, the study provides a comparative analysis of System 1 versus System 2 reasoning techniques for action planning. Finally, we introduce a disentangled evaluation methodology that isolates cognitive plan quality from retrieval performance, establishing a robust benchmark for task completeness and parameter accuracy in knowledge-intensive industrial applications.

2. Related Works

Agentic LLMs unify reasoning, acting, and interacting into dynamic procedures centered around action planning. To optimize this reasoning step when addressing user queries, various structured architectures have been developed with distinct operational philosophies. These range from ReAct, which employs a continuous loop of reasoning, acting, and retrieving observations, to ReWOO [Xu et al. 2023], which strictly decouples the upfront generation of a planning to-do list from its subsequent execution by a worker node. Further enhancing planning robustness, MultiPlan generates diverse plan variations—adjusting prompts or temperatures—and utilizes a decider to select the optimal trajectory for Self-Consistency [Wang et al. 2023]. Finally, while systems like Reflexion [Shinn et al. 2023] and Self-Refine [Madaan et al. 2023] integrate planning and retrieval within iterative, self-improving loops, other frameworks such as SwiftSage [Lin et al. 2023] introduce alternative methodologies designed to emulate the underlying cognitive processes of human planning.

Effective tool utilization is another critical component of agentic systems, with recent taxonomies evaluating an agent's proficiency based on its capacity to determine when, which, and how to invoke specific tools. To enhance this accuracy, established techniques like Few-Shot and Dynamic Few-Shot prompting provide explicit execution examples, while Multi-Tool CoT [Inaba et al. 2023] fosters deliberative reasoning by illustrating the requisite cognitive process. For greater reliability, more complex architectures deploy advanced validation mechanisms—such as ToolVerifier [Mekala et al. 2024]—which introduce dedicated nodes to strictly guarantee the accurate selection and parameterization of tools prior to execution.

With the advance of Planning and Tool Use techniques, arises the necessity of a method to evaluate an agent's ability to perform them. There exist many frameworks which achieve that goal. Some of them use more traditional methods, such as semantic similarity to a verified output, and other frameworks use evaluation methods such as LLM-as-a-Judge to score plans, tool calling, and answers.

3. Methodology

3.1. Proposed Method

The proposed method implements a hierarchical multi-agent architecture designed to decouple high-level strategic orchestration from specialized technical execution. The system is composed of an LLM Supervisor acting as the central reasoning engine and four domain-specialized sub-agents, as presented in Figure 1. This functional decoupling allows for the parallel processing of sub-tasks and the application of domain-specific logic. The architecture integrates external capabilities through two tools.

The LLM Supervisor functions as the primary brain of the system, responsible for decomposing user queries into sub-tasks and selecting the appropriate sub-agent or tool. To optimize the planning phase, the study evaluates a spectrum of reasoning paradigms. While the baseline is established using the reactive ReAct model, the methodology compares it against structured and deliberative techniques such as ReWOO [Xu et al. 2023] and MultiPlan [Wang et al. 2023]. Furthermore, the system tests advanced deliberative modules like SwiftSage [Lin et al. 2023], alongside iterative self-correction loops such as Reflexion [Shinn et al. 2023] and Self-Refine [Madaan et al. 2023]. To maximize retrieval accuracy across complex datasets, every sub-agent implements an Advanced RAG pipeline. This process begins with Query Augmentation, specifically utilizing Step Back Prompting [Zheng et al. 2024] to abstract the user's initial request into fundamental concepts and synonym. Following the retrieval phase, the sub-agents execute a Reranking stage using RankGPT [Sun et al. 2023].

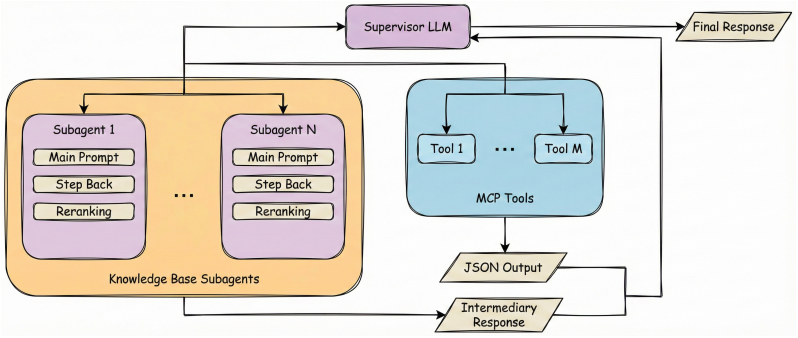

Figura 1. Proposed hierarchical multi-agent architecture for Agentic RAG. The system is governed by a Supervisor LLM that serves as the primary reasoning engine. On the left, the Knowledge Base Subagents execute an Advanced RAG pipeline. Concurrently, the tools block integrates external capabilities through Model Context Protocol. The Supervisor LLM synthesizes these intermediary responses and tool data to generate a final response.

3.2. Experiments

To address industrial data complexities, the proposed hierarchical architecture implemented via the LangGraph and DeepAgents frameworks employs a GPT-4.1 LLM Supervisor to orchestrate tasks. The sub-agents navigate a proprietary consulting knowledge base utilizing Chroma as a vector database ($K = 3$). The system's System 1 and System 2 reasoning capabilities were tested against an expert-validated dataset of human-annotated corporate queries, which was stratified into four distinct complexity levels: Direct Factual Retrieval, Tool-Dependent Queries requiring precise external parameter matching, Complex Synthesis and Onboarding for high-level multi-document aggregation, and Instruction-Constrained Queries evaluating strict adherence to formatting and negative constraints.

To provide an overview of the evaluation data without compromising corporate confidentiality, the data profile is detailed in aggregated form. The proprietary dataset comprises over 2,000 documents and structured records spanning corporate knowledge management, consulting methodologies, and internal human resources directories. The multimodal corpus includes unstructured instructional materials (e.g., strategic planning

frameworks, onboarding guides), semi-structured data (e.g., operational KPI spreadsheets), and structured employee databases. In total, the knowledge base contains approximately 2.4 million tokens, with an average document length of 1,135 tokens. This heterogeneity natively tests the agents' ability to reason across abstract methodological concepts and exact database registries. The evaluation itself was conducted using 350 curated test queries, averaging 112 words per prompt.

Furthermore, while benchmarks offer standardized evaluation grounds, they frequently fail to capture the structural heterogeneity, domain-specific jargon, and operational noise inherent to real-world corporate environments. This study deliberately employs a proprietary industrial dataset to evaluate Agentic RAG because it preserves the complexity of enterprise tasks. By doing so, this research presents a real-world use case that academic datasets currently cannot fully replicate. The evaluation of this architecture on benchmarks is earmarked as a promising direction for future work.

To isolate cognitive planning from technical execution, the evaluation employs four granular metrics. Plan Quality assesses the reasoning module's logical coherence and efficiency via G-Eval [Liu et al. 2023] utilizing GPT-5.1. Success Rate measures end-to-end task completeness, evaluated by two trained and calibrated human annotators with access to the ground truth. Technical execution is evaluated through Tool Correctness, which validates the appropriate selection of tools or sub-agents, and Argument Correctness, which verifies the syntactic precision of the parameters passed; both are calculated via exact match against expert-provided ground truths.

4. Results

The experimental results, summarized in Table 1, provide an evaluation of the proposed hierarchical architecture across various planning and tool-use combinations. When analyzing the Standard Tool Use baseline, the results reveal a gap between abstract reasoning and technical execution. SwiftSage achieved the highest Plan Quality at 42.24%, yet yielded the lowest Success Rate (73.33%) and Argument Correctness (50.69%) in this category. This suggests that while System 2 deliberate planning generates sophisticated task decompositions, it struggles with the syntactic precision required for tool integration without external validation or explicit examples, especially when System 1 is activated. In contrast, the ReWOO framework demonstrated superior robustness by decoupling the planning phase from the observation and execution phases. Under Standard Tool Use, ReWOO maintained a high Tool Correctness of 93.33% despite a lower initial Plan Quality of 15.83%. This indicates that modular plans, although less complex in their initial sequence, are more resilient to execution errors in the multi-hop workflows typical of the consultancy dataset.

The introduction of the Few-Shot strategy provided the most consistent performance gains across the different planning techniques. Specifically, the combination of the ReWOO and Few-Shot examples achieved the maximum success rate of 93.33%. This configuration bridges the gap between the supervisor's orchestration and the technical requirements of the tools, proving that explicit execution patterns are vital for agents interacting with specialized knowledge bases.

Interestingly, iterative planning techniques such as Reflexion and Self-Refine showed a slight performance degradation in Success Rate when moving from Standard

Tabela 1. Comparative Analysis of Planning and Tool Use Techniques in Agentic RAG

Planning Technique	Tool Use Technique	Plan Quality (%)	Success Rate (%)	Arg. Correctness (%)	Tool Correctness (%)
Block 1: Variable Planning with Standard Tool Use					
ReAct	Standard	—	83.33	80.56	83.33
ReWOO	Standard	15.83	83.33	90.23	93.33
Reflexion	Standard	—	90.00	80.56	83.33
Self Refine	Standard	25.17	90.00	80.46	83.33
Swift Sage	Standard	42.24	73.33	50.69	70.00
Multiplan	Standard	28.33	83.33	77.78	83.33
Block 2: Variable Planning with Few-Shot Tool Use					
ReAct	Few Shot	—	86.67	80.56	83.33
ReWOO	Few Shot	16.67	93.33	80.89	86.67
Reflexion	Few Shot	—	86.67	88.50	90.00
Self Refine	Few Shot	27.00	76.67	84.00	90.00
Swift Sage	Few Shot	31.33	80.00	56.88	73.33
Multiplan	Few Shot	26.67	83.33	73.19	83.33
Block 3: Standard Planning with Variable Tool Use Techniques					
Standard	Few Shot	—	80.00	74.07	80.00
Standard	Dynamic Few Shot	—	90.00	77.71	86.67
Standard	MultiTool CoT	—	86.67	78.00	90.00
Standard	ToolVerifier	—	86.67	90.00	93.33
Block 4: Variable Planning with ToolVerifier					
ReAct	ToolVerifier	—	86.67	80.56	83.33
ReWOO	ToolVerifier	15.33	90.00	81.00	90.00
Reflexion	ToolVerifier	—	86.67	83.78	86.67
Self Refine	ToolVerifier	24.48	83.33	77.71	86.67
Swift Sage	ToolVerifier	42.00	83.33	83.78	86.67
Multiplan	ToolVerifier	24.17	90.00	84.00	93.33

to Few-Shot tool use. For example, Reflexion's Success Rate dropped from 90.00% to 86.67%. This phenomenon suggests a potential cognitive conflict within the LLM, where the provided examples may interfere with the agent's internal self-correction loops, leading to sub-optimal refinements.

The evaluation of standalone tool-use techniques, independent of advanced planning, highlights the efficacy of context-aware selection. Dynamic Few-Shot achieved a 90.00% Success Rate, significantly outperforming the static Few-Shot baseline of 80.00%. By selecting examples based on similarity to the current query, the system effectively mitigates technical hallucinations and optimizes the selection of tools in low-similarity contexts.

The ToolVerifier module emerged as an essential reliability layer for the most complex planning strategies. SwiftSage's Argument Correctness surged from 50.69% in the Standard configuration to 83.78% when the ToolVerifier was active. This additional validation node ensures that the arguments passed to MCP tools adhere to the correct schema and ranges before execution, which is critical for maintaining system integrity in industrial applications.

The Multiplan (Self-Consistency) approach also benefited significantly from the integration of the ToolVerifier, demonstrating marked improvements over its standard baseline across all technical metrics. When compared to the standard configuration, the Success Rate rose from 83.33% to 90.00%, while Argument Correctness increased from 77.78% to 84.00%, and Tool Correctness improved from 83.33% to 93.33%. These gains demonstrate that generating multiple reasoning paths and subsequently verifying each

step through a dedicated node is a highly effective strategy for ensuring task completeness and technical precision in high-stakes industrial environments

Across all experiments, Tool Correctness remained notably high, frequently exceeding 83%. This indicates that the LLM Supervisor is highly effective at identifying the appropriate tools even when the planning logic varies. However, the lower Argument Correctness in several SwiftSage and Multiplan runs underscores that parameter precision is a more frequent point of failure than tool selection itself in multi-agent RAG systems.

Overall, the results for Plan Quality, exhibit a lower order of magnitude across all test configurations compared to other performance metrics. This conservative scoring is attributed to the high degree of rigor applied by the judge node when evaluating the logic and complexity of planned action sequences. Despite these lower absolute numerical values, the findings remain internally consistent and logically sound, as the relative variations in Plan Quality accurately reflect the cognitive depth of the different paradigms. For instance, while SwiftSage reached the highest quality score of 42.24%, the overall system effectiveness is validated by the fact that even modular plans led to successful outcomes, with Success Rates and Tool Correctness frequently exceeding 83%.

Finally, the results confirm that the proposed hierarchical architecture performs best when combining modular planning (ReWOO) with explicit execution guidance. While deliberate planning provides high-quality blueprints, the inclusion of a ToolVerifier or Dynamic Few-Shot mechanism is necessary to reach the reliability levels required for processing the complex datasets found in industrial use cases.

5. Conclusions

This research validates the efficacy of a hierarchical multi-agent architecture within an Agentic RAG framework, demonstrating that decoupling strategic orchestration from technical execution enhances robustness in complex industrial environments. A comparative analysis of reasoning paradigms revealed a critical trade-off: while deliberative System 2 planning (e.g., SwiftSage) yields superior logical plan quality, it frequently falters in the syntactic parameter precision required for reliable tool integration. Conversely, the structured, modular approach of ReWOO combined with Few-Shot guidance achieved a peak success rate of 93.33%, indicating that for data-rich enterprise applications, modular execution is more resilient to error propagation than iterative reasoning loops.

To bridge the gap between high-level planning and precise technical execution, the implementation of a verification mechanisms proved essential. Deploying a ToolVerifier module successfully mitigated technical hallucinations, improving SwiftSage's argument correctness from 50.69% to 83.78%, while context-aware Dynamic Few-Shot techniques significantly outperformed static baselines. These results justify the proposed disentangled evaluation methodology, which traces system failures to either cognitive missteps or specific technical errors. Looking forward, future research should investigate the architecture's scalability, explore real-time autonomous adaptation of sub-agents to reduce computational overhead, and resolve cognitive conflicts between iterative self-correction loops and few-shot examples to further stabilize industrial applications.

Referências

Gao, S., Zhao, S., Jiang, X., Duan, L., Chng, Y. X., Chen, Q.-G., Luo, W., Zhang, K., Bian, J.-W., and Gong, M. (2025). Scaling beyond context: A survey of multimodal retrieval-augmented generation for document understanding. *arXiv preprint arXiv:2510.15253*.

Inaba, T., Kiyomaru, H., Cheng, F., and Kurohashi, S. (2023). Multitool-cot: Gpt-3 can use multiple external tools with chain of thought prompting. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL 2023)*, pages 1522–1532. Association for Computational Linguistics.

Kahneman, D. (2011). *Thinking, Fast and Slow*. Farrar, Straus and Giroux.

Lin, Y., Cheng, Z., Zhao, A., Chen, S., Fu, G., Zhang, S., and Wu, D. (2023). Swiftsage: A modular agent with swift and sage components for adaptive and efficient action. In *Advances in Neural Information Processing Systems (NeurIPS 2023)*.

Liu, Y., Iter, D., Xu, Y., Wang, S., Xu, R., and Zhu, C. (2023). G-eval: Nlg evaluation using gpt-4 with better alignment than human annotation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP 2023)*, pages 10654–10671. Association for Computational Linguistics.

Madaan, A., Tandon, N., Gupta, P., Hallinan, S., Gao, L., Wiegreffe, W., Alon, U., Dziri, N., Prabhumoye, S., Yang, Y., Gupta, S., Majumder, B. P., Lapania, G., Welleck, S., and Bhadauria, T. (2023). Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems (NeurIPS 2023)*.

Mekala, D., Weston, J., Lanchantin, J., Raileanu, R., Lomeli, M., Shang, J., and Dwivedi-Yu, J. (2024). Toolverifier: Generalization to new tools via self-verification. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 5026–5041, Miami, Florida, USA. Association for Computational Linguistics.

Shinn, N., Cassano, F., Berman, E., Gopinath, A., Narasimhan, K., and Yao, S. (2023). Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS 2023)*.

Sun, W., Yan, L., Ma, X., Wang, S., Ren, P., Chen, Z., Yin, D., and Ren, Z. (2023). Is chatgpt good at search? investigating large language models as re-ranking agents. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP 2023)*.

Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E., Narang, S., Chowdhery, A., and Zhou, D. (2023). Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations (ICLR 2023)*.

Xu, B., Peng, Z., Lei, B., Mukherjee, S., Liu, Y., and Xu, D. (2023). Rewoo: Decoupling reasoning from observations for efficient augmented language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP 2023)*.

Zheng, H. S., Mishra, S., Chen, X., Cheng, H.-T., Chi, E. H., Le, Q. V., and Zhou, D. (2024). Take a step back: Evoking reasoning via abstraction in large language models. In *International Conference on Learning Representations (ICLR)*.