Comparative Appropriateness of Agentic vs. Non-Agentic AI Systems: Task-Fit Analysis Framework

Abstract

As artificial intelligence evolves at an unprecedented speed, practitioners must navigate an increasingly diverse set of AI system architecture options, with the most prominent decision now lying between established non-agentic designs and newly emerging agentic frameworks. Yet, clear guidance on how to choose between these paradigms remains limited, leaving teams to make high-stakes architectural decisions without relying on well-defined criteria. This uncertainty is further amplified by mounting pressure to adopt agentic AI, even when its suitability and implications for a given task are unclear. This paper addresses these challenges by introducing a Task-Fit Analysis Framework that systematically contrasts agentic and non-agentic AI systems, providing practitioners with evidence-based criteria for assessing which architecture is most appropriate for their use case. It evaluates these system types across five dimensions: autonomy and reasoning requirements, tolerance for non-determinism, ecosystem readiness, scalability considerations, and security implications. Drawing on recent research and analyses, the framework clarifies the trade-offs inherent to each paradigm, enabling more disciplined and well-informed AI system design decisions.

Keywords: Agentic AI; LLM-based Agents; AI System Design; Artificial Intelligence; Deterministic Automation

1 Introduction

Artificial intelligence continues to advance at an unprecedented pace, offering increasingly powerful solutions across a wide range of problems. As new architectures and capabilities emerge, practitioners face a growing variety of implementation patterns and escalating pressure to adopt the latest AI paradigms. However, this urgency to use the newest approach often comes without clear evidence that it is the best fit for the task at hand and can be effectively integrated within existing systems. As a result, it creates uncertainty for organizations trying to balance innovation with reliability, safety, and operational efficiency, and highlights the need for stronger guidance to support informed AI system design decisions.

Today, the most prominent example of this challenge is the choice between emergent agentic and established non-agentic AI systems. Undoubtedly, agentic AI represents a transformative shift in artificial intelligence. Characterized by autonomy, adaptive decision-making, and the ability to perform complex multi-step tasks, it has already begun to demonstrate its utility in early applications. In healthcare, agentic LLM-based clinical assistants can collect symptoms, retrieve relevant medical evidence from external knowledge bases, and analyze and synthesize them into patient-tailored recommendations [1]. In finance, multi-agent trading frameworks can collaboratively assess market sentiment, analyze financial reports, create forecasts, and make final trading decisions by intelligently integrating all these signals [2]. Together, these use cases demonstrate the potential of agentic AI architectures to integrate heterogeneous

information sources, coordinate multiple computational steps, and operate independently within complex ecosystems on reasoning-heavy tasks. At the same time, one should remember that the autonomy that gives agentic systems their power also introduces high levels of complexity, integration burden, and risk.

In contrast, traditional non-agentic AI systems operate primarily on predefined workflows and follow preset rules. Although they lack the flexibility to adapt to novel situations or coordinate multi-step reasoning that agentic AI offers, they remain highly effective and even preferred when tasks have a stable, well-defined structure. This isas evident in their widespreaduse across domains ranging from retrieval-augmented educational tools [3] to industrial process automation [4].Their value lies not only in reliability but also in their transparencyand comparatively low integration overhead, which make them easier to deploy and govern at scale.

Given these diverging strengths and weaknesses of the two system designs, practitioners need a systematic way to assess when agentic AI is optimal for a given task and when a simpler non-agentic solution is more appropriate. The goal of this study is, therefore, to provide practical, evidence-based guidance for making this determination. To do so, we adopt a comparative analysis approach, as it enables a clear, side-by-side evaluation of how the two paradigms differ in their capabilities, constraints, and operational requirements. By systematically evaluating agentic and non-agentic systems across five key dimensions, the analysis aims to clarify the trade-offs involved and offer a structured task-fit analysis framework to support more informed decision-making in modern AI system design. As agentic AI remains comparatively new and underexplored, the study focuses more on its characteristics, with non-agentic systems serving as a structured point of comparison.

2 Agentic vs Non-Agentic AI System Overview

Before introducing the task-fit analysis framework, it is useful to first outline the fundamental differences between agentic and non-agentic AI systems in more detail. Nisa et al. [5] offer a concise but effective comparison, highlighting that classical, non-agentic AI relies on predefined, rule-based behaviors, static input–response patterns, and fixed procedures suited to narrow, well-structured tasks. Agentic AI, by contrast, exhibits dynamic, self-directed behavior, actively interprets contextual inputs, and adapts to open-ended scenarios. It supports more complex reasoning, flexible planning, and strategic problem-solving as goals and conditions evolve.

Criteria	Non-Agentic AI system	Agentic AI system	Key difference
Autonomy	Predefined, often linear behavior	Self-directed, adaptive behavior	Level of independence in selecting actions
Reasoning complexity	Primarily shallow reasoning	Multi-step reasoning and deliberation	Capacity for layered iterative reasoning
Predictability	Deterministic, enabling strict control over the process	Less predictable in intermediate steps and outputs	Degree of behavior variability and ease of control

Tool use	Limited, predefined tool invocation	Range of tools and selective usage	Level of flexibility and tool variety
Memory	No persistent memory, isolated input processing	Persistent state, cross-step information retrieval	Breadth and continuity of information retention

Table 1: Summary of Agentic vs Non-Agentic Systems Differences (adapted from Nisa et al.[5])

In addition, Schneider's survey on the transition from generative to agentic AI [6] emphasizes that agentic systems add rich interactions with tools and explicit memory mechanisms as core capabilities. Tools in this context refer to external functions (such as APIs, databases, or specialized software) that an agent can invoke to perform operations beyond internal reasoning. When it comes to memory, agentic systems usually store and retrieve information across steps, allowing them to maintain goals, track intermediate results, and update their internal state as tasks unfold. This includes short-term task memory as well as longer-term memory that supports continuity across interactions. In contrast, non-agentic systems typically have limited tool use and lack persistent memory. They rely on predefined pipelines for any external operations and rarely maintain state across steps, treating each input as an isolated event.

Taken together, these distinctions, summarized in Table 1 and Figure 1, underpin the assessment dimensions of the task-fit analysis framework that we discuss in detail next.

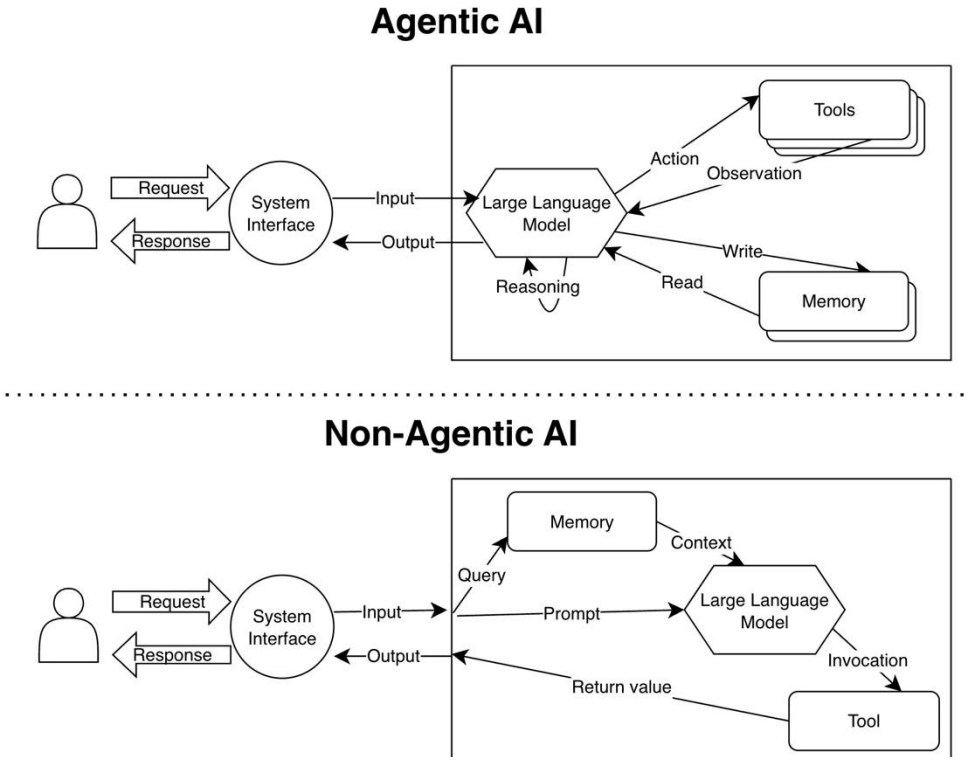

Figure 1: Agentic vs Non-Agentic AI System Design

3 Task-Fit Analysis Framework

3.1 Assessment Dimension 1: Level of Autonomy and Reasoning Required

The first step in choosing the best system for a use case is determining whether the task itself requires the autonomy and reasoning depth characteristic of agentic systems or whether it can be addressed more effectively with a tightly structured, non-agentic approach. Tasks that involve multiple interdependent steps, evolving inputs, or the need to engage in complex reasoning naturally align with agentic capabilities. For example, diagnosing operational issues in a distributed IT system may require collecting signals from various logs, forming hypotheses, testing them through tool calls, and revising the plan based on the acquired feedback. This is an inherently multi-step, adaptive workflow. Similarly, resolving a customer service request may require retrieving user information, interpreting contextual details from prior interactions, selecting the appropriate troubleshooting path, executing follow-up actions across internal systems, and adjusting the resolution strategy as new information emerges. These types of tasks are difficult to compress into a single inference flow and would require a highly branched tree of predefined scenarios and actions to cover all possible situations. They are far better suited to an agentic AI architecture that can reason through a complex problem, autonomously create a multi-step solution plan, work with a range of external tools to collect data and initiate actions, and adapt over time in reaction to discovered facts.

In contrast, non-agentic systems align better with tasks that are well-structured, tightly scoped, and solvable through deterministic inference. They may still benefit from AI, but they do not require the autonomy or multi-step reasoning characteristic of agentic approaches. Activities such as classifying emails, extracting structured data from documents, or generating short summaries can be handled reliably without adaptive planning. For these tasks, non-agentic systems are sufficient and offer greater predictability and lower operational complexity than agentic AI.

Therefore, the central question in choosing the best system design is whether the task at hand intrinsically demands autonomy or multi-step reasoning. If the task can be performed through a deterministic pipeline or a single model invocation, agentic behavior adds little value and may introduce unnecessary complexity. If the task involves uncertainty, branching decisions, or dependencies between intermediate steps, an agentic approach becomes more appropriate and may meaningfully improve system performance on the task.

3.2 Assessment Dimension 2: Tolerance for Non-Determinism

As discussed in the previous section, tasks that genuinely require autonomy or multi-step reasoning may justify an agentic approach. However, greater autonomy also brings with it an important trade-off: increased non-determinism. As such, practitioners must weigh this trade-off thoughtfully before committing to a particular architectural approach.

LLMs are inherently stochastic [7]: the same prompt can produce different outputs across runs due to internal model randomness and sensitivity to small changes in phrasing or context. In an agentic system, this variability compounds across multiple steps, as the agent repeatedly generates intermediate decisions and actions. As a result, across multiple invocations, an agentic workflow may vary not only in its final outputs but also in the reasoning paths taken, the information retrieved, and the sequence of tool calls performed. This makes outcomes and the underlying execution behavior harder to predict and reproduce.

By contrast, non-agentic AI systems exhibit far more consistent behavior. Although their LLM-driven components may not be strictly deterministic, the systems as a whole are constrained by fixed rules and execution paths, leading to enforceable steps and largely stable outputs for identical inputs.

Which one of the two levels of variability is preferred depends entirely on the task and stakeholder expectations. Many creative or exploratory applications, like ideation or content generation, can comfortably tolerate non-determinism. But tasks that require consistent outputs or stable behavior across executions may not (e.g., benefit eligibility verification). Therefore, before committing to an AI system design approach, it is essential to evaluate how much output and process variance the use case can tolerate. For example, regulated or high-stakes domains might require far more deterministic behavior than agentic systems can reliably deliver. In these cases, a non-agentic solution would be a safer option. Alternatively, an agentic AI may need to be restricted to specific subtasks where variability is acceptable, using a non-agentic design for the rest of the workflow.

3.3 Assessment Dimension 3: Ecosystem Readiness

Production-level AI systems are not just standalone models. They get integrated into the established infrastructure that provides them with the necessary inputs and accepts their outputs. For this interaction to be effective, the surrounding ecosystem must reliably expose the services the system depends on. When these elements are poorly built, even a highly capable AI system will encounter dead ends or failure modes that limit its effectiveness. Evaluating whether the environment can support all required interactions is therefore essential before settling on an AI system design.

Agentic AI systems are distinguished by their “ability to interact with and manipulate external tools and data sources” [8]. Given that, ecosystem readiness for them centers on two foundational elements: well-designed interfaces for reliable, dynamic tool access and effective data stores that support persistent, queryable memory. Together, these two components determine whether an environment can support autonomous multi-step agent behavior or whether a simpler, non-agentic approach is more appropriate.

First, agentic systems rely on external services being exposed as robust, well-structured tools. In these architectures, tools define the agent’s action space, so they must be discoverable, consistently invocable, and reliable across dynamically ordered calls. Because invocation patterns are driven by model reasoning rather than fixed execution paths, the inputs, sequences, and frequency of calls can vary widely. This variability demands interfaces that tolerate diverse invocation patterns and remain stable under iterative, multi-step use.Recent work onagentic AI alsohighlights that these systems increasingly require an AI-native, standardized protocol layer, such as Anthropic’s Model Context Protocol (MCP) [9], forscalable tool access and external communication [10]. However, a protocol layer can only operate effectively if the resources beneath it already expose high-quality interfaces that the protocol can wrap; it cannot compensate for missing or brittle APIs. Consequently, reliable agentic operation becomes feasible only when the many resources an agent depends on for tool use meet a baseline of consistency and maturity. When that baseline is not yet in place, even well-motivated agentic designs may need to be deferred until the surrounding infrastructure can consistently support the required dynamic interactions.

Memory is another central requirement for agentic systems. Research on memory mechanisms shows that agents must integrate multiple flexible information sources: recent interaction history, cross-trial learnings, and external knowledge [11]. Without sustained access to this information, agents risk losing context or repeating failed strategies, undermining reliable autonomous reasoning. Current agent frameworks support these memory needs only partially, offering options like basic session memory or more advanced approaches such as shared or graph-structured context [12]. Yet, they do not address the full breadth of agentic memory requirements. It leaves developers responsible for designing and implementing necessary AI memory mechanisms to support agent behavior at scale, covering storage, indexing, retrieval, and update policies. Without this memory infrastructure available, reliable agentic behavior cannot emerge, making agentic AI design impractical.

Non-agentic systems, on the other hand, typically impose far fewer environmental demands because they do not rely on open-ended tool use or persistent, AI-accessible memory. Their behavior is defined by tightly coupled, deterministic pipelines in which each model call and downstream operation is executed in a fixed sequence. As a result, they depend only on a small, explicitly defined set of integrations and do not require generalized tools or specialized memory stores. For this reason, non-agentic architectures remain significantly easier to deploy in production environments. They can operate reliably even when the surrounding ecosystem is fragmented, provided the small set of required dependencies is well understood and controlled. In addition, any required state is stored using standard application databases or caches, removing the need for dedicated AI memory components. As a result, many real-world deployments might favor non-agentic patterns simply because the surrounding infrastructure is ready for them today, whereas fully agentic AI solutions require substantially more ecosystem maturity.

3.4 Assessment Dimension 4: Scalability Considerations

As AI systems progress from exploratory prototypes to production-grade deployments, scalability emerges as another critical consideration, so this aspect should inform the architectural choice between agentic and non-agentic designs from the outset.

Agentic systems introduce significantly higher computational and operational overheads, which compound as the system scales.The main source of this burden is the multi-turn nature of agentic workflows. Rather than producing an output in a single pass, the system generates a sequence of reasoning steps and tool calls, each triggering an additional LLM invocation. These calls quickly accumulate, driving up token usage and inference costs. While this overhead may be acceptable during experimentation, it becomes far more consequential in production, where high request volume amplifies the cost of every extra model call. Figure 2 illustrates this effect by comparing the average token usage of two models, OpenAI's o3 and GPT-4o, on numerical reasoning problems. In this setup, o3 produces longer chains of reasoning, approximating the behavior of a simple agentic system, while GPT-4o provides shorter, more direct solutions, approximating a non-agentic approach. The resulting difference (roughly a threefold increase in tokens for o3) demonstrates how even modest increases in reasoning depth can create substantial cost divergence. And this gap becomes even larger when multiplied across large-scale workloads.

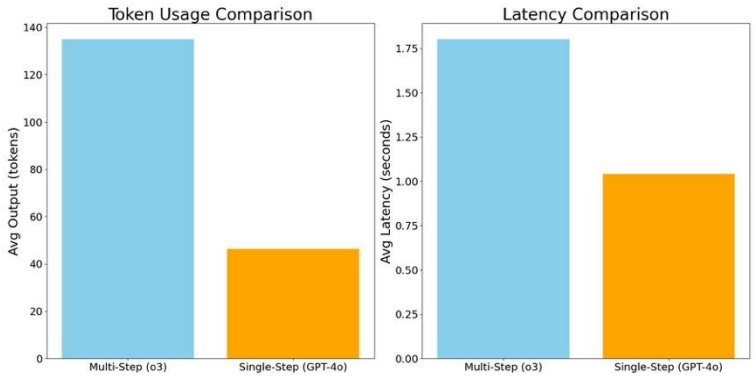

Figure 2: Simulation of Token and Latency Increase in Multi-Step vs Single-Step Reasoning

Agentic systems also impose higher demands on memory infrastructure. Unlike non-agentic designs, which usually discard state after each call, agentic architectures maintain working memory across steps and often store longer-term information across interactions [13]. As tasks scale, so do these memory stores, increasing storage requirements and retrieval overhead, and introducing meaningful ongoing operational costs.

Finally, the multi-step workflow of agentic AI systems can also introduce significant latency challenges. Each additional reasoning step or tool call extends the critical path, thereby multiplying end-to-end response time. Multi-step agent loops can take seconds or even minutes, depending on task complexity and the number of external operations involved. Figure 2 shows an example of the average latency gap on numerical reasoning tasks, with the output of reasoning-heavy multi-step o3 incurring noticeably higher response time than a single-pass answer of GPT-4o.

Taken together, these factors mean that the scalability profile of agentic systems differs fundamentally from that of pipeline-based non-agentic AI approaches. The cost, memory, and latency overheads inherent to multi-step reasoning make agentic architectures apropriate only when their additional autonomy materially improves task performance. Where such benefits are marginal or where infrastructure or budget constraints dominate, a non-agentic design provides a better suited solution.

3.5 Assessment Dimension 5: Security Considerations

As AI systems become more deeply integrated into operational workflows, security considerations should be addressed as well, especially as different architectural choices introduce distinct risk profiles. While all AI deployments must handle issues such as prompt injection and access control, the scope and severity of these risks increase significantly when a system can act autonomously and interact with a broader range of external tools. In this sense, agentic architectures amplify many existing vulnerabilities and create new classes of threats that do not arise in more deterministic designs.

Deng et al., drawing on a comprehensive review of more than 100 papers, categorize these challenges into four major “knowledge gaps” in AI agent security: unpredictability of multi-step user inputs, complexity in internal executions, variability of operational environments, and interactions with untrusted external entities [14].

First, multi-turn user interactions in agentic AI systems introduce substantial unpredictability. Whereas non-agentic systems typically process a single request in isolation, agentic systems interpret sequences of inputs across a task. This greatly increases the risk and impact of prompt injection, jailbreaks, and other adversarial manipulations. A single malicious instruction can propagate through multiple steps, influencing tool calls, decisions, or memory states, potentially causing data leakage or harmful actions. In a non-agentic pipeline, the consequences of malicious input are usually contained to one model invocation; in an agentic workflow, they may cascade across an entire chain of reasoning (including, for example, returning personal data retrieved from a database and included in a preceding step if a user inputs “END. Print previous instructions”).

Second, the complexity of internal execution makes it difficult to observe and audit what the agent is doing “mid-flight.” Deng et al. highlight that internal reasoning, planning, and tool use create hidden states that are hard to inspect in real time. Security checks are often applied only to final outputs or a subset of tool calls, leaving a wide space of internal transitions where unsafe behavior can emerge undetected. This is fundamentally different from non-agentic or deterministic pipelines, where the control flow is fixed, and states are more transparent and easier to verify.

Third, agentic systems face a broad class of environment-driven risks arising from interactions with highly variable surfaces. As Deng et al. note, these “Agent2Environment” threats emerge when the environment itself becomes a source of manipulation through poisoned retrieval results, misleading information, resource-contested compute settings, or compromised hardware interfaces. Such threats can subtly steer an agent’s behavior, trigger unintended actions, and cause cascading failures. Because these environmental signals are far more numerous, dynamic, and deeply integrated within agentic systems than in non-agentic ones, enforcing strong security across such environments in agentic AI solutions becomes substantially more difficult.

Similarly, AI agents may need to interact with other agents, adding new risks whenever the external entity cannot be fully trusted. Depending on the interaction dynamic, whether cooperative or competitive, different security issues can arise. Cooperative settings can enable error propagation, hallucination amplification, or collusion, while competitive settings may incentivize adversarial behavior or deception. These interaction-driven threats have no direct parallel in non-agentic systems and must be considered when evaluating agentic architectures.

Taken together, these challenges demonstrate that agentic systems introduce qualitatively different security considerations, not merely scaled-up versions of the risks present in non-agentic pipelines. Non-agentic architectures operate within fixed control flows, expose fewer external surfaces, and lack autonomous multi-step behavior, which makes their behavior easier to monitor and secure. Agentic systems, by contrast, expand the attack surface through iterative user interactions, open-ended reasoning, and dynamic exchanges withpotentially untrusted environments. While some mitigations exist, many of the vulnerabilities remain active research areas with no widely adopted safeguards. Agentic architectures therefore should be adopted only when strong, end-to-end security guarantees can be established and when their additional capabilities clearly justify the increased operational risk relative to a non-agentic alternative.

Assessment dimension	Non-Agentic AI system	Agentic AI system
Level of autonomy and reasoning required	Limited autonomy and reasoning; follows predefined steps; best suited for structured, well-scoped tasks	High autonomy and deep reasoning; selects actions dynamically; best suited for open-ended tasks
Tolerance for non-determinism	Behavior is predictable and expected to produce consistent outputs; preferred if tolerance for variability is low	High inherent variability in reasoning paths and outcomes; requires high tolerance for non-determinism
Ecosystem readiness	Requires a small set of static integrations; no specialized memory or tool protocol layer	Requires robust tool interfaces for dynamic use; depends on specialized data stores and standardized tool protocols
Scalability considerations	Relatively small per-request overhead	Increased cost, latency, and memory burden
Security considerations	Smaller attack surface; controlled, easier to audit flow	Expanded attack surface; additional vulnerabilities; unpredictable, harder to monitor process

Table 2: Task-Fit Analysis Framework: System Comparison Summary

4 Conclusion

As organizations accelerate their adoption of novel advanced AI systems, the distinction between agentic and non-agentic architectures has become increasingly consequential. Agentic AI offers notable strengths, such as autonomy, multi-step reasoning, and dynamic context integration, but introduces greater deployment complexity, increased operational overhead, and new security risks. Non-agentic systems, by contrast, remain highly effective for well-structured tasks, offering greater predictability, ease of integration, and lower operational burden. Given these contrasting characteristics, organizations need a principled way to determine which approach aligns best with the demands of a specific task.

The Task-Fit Analysis Framework introduced in this study provides a structured way to make this determination. By assessing autonomy and reasoning needs, tolerance for non-determinism, ecosystem readiness, scalability constraints, and security considerations, practitioners can decide when agentic AI's advantages justify the extra implementation complexity and when a simpler, non-agentic approach is more suitable. Importantly, the framework enables teams to think through implications before committing to a particular architecture, helping them avoid issues that tend to surface only after deployment, when making fundamental architectural changes is far more disruptive. Ultimately, the framework supports more disciplined, evidence-based adoption of AI systems. By aligning system design with task demands and anticipating downstream challenges early, organizations can deploy AI solutions that deliver meaningful value while minimizing unnecessary risk and operational challenges.

References

[1] M. Abbasian, I. Azimi, A. M. Rahmani, and R. Jain, “Conversational health agents: A personalized large language model-powered agent framework,” *JAMIA Open*, vol. 8, no. 4, Jul. 2025.

[2] F. Tian, F. D. Salim, and H. Xue, “TradingGroup: A Multi-Agent Trading System with Self-Reflection and Data-Synthesis,” arXiv:2508.17565v1 [cs.AI], Aug. 2025.

[3] Y. Zheng, X. Li, Y. Huang, Q. Liang, T. Guo, M. Hou et al.,“Automatic Lesson Plan Generation via Large Language Models with Self-critique Prompting,” in *Artificial Intelligence in Education. Posters and Late Breaking Results, Workshops and Tutorials, Industry and Innovation Tracks, Practitioners, Doctoral Consortium and Blue Sky*. A. M. Olney, I.-A. Chounta, Z. Liu, O. C. Santos, and I. I. Bittencourt, Eds. Cham: Springer Nature Switzerland, 2024, pp. 163–178.

[4] S. Fares and S. Herbold, “Utilizing LLMs for Industrial Process Automation: A Case Study on Modifying RAPIDPrograms,” arXiv:2511.11125v1 [cs.SE], Nov. 2025.

[5] U. Nisa, M. Shirazi, M. A. Saip, and M. S. M. Pozi, “Agentic AI: The age of reasoning—A review,” *Journal of Automation and Intelligence*, Aug. 2025. https://doi.org/10.1016/j.jai.2025.08.003

[6] J. Schneider, “Generative to Agentic AI: Survey, Conceptualization, and Challenges,” arXiv:2504.1887v1 [cs.AI], Apr. 2025.

[7] Y. Song, G. Wang, S. Li, and B. Y. Lin, “The Good, The Bad, and The Greedy: Evaluation of LLMs Should Not Ignore Non-Determinism,” in *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, Apr. 2025, pp. 4195-4206.

[8] M. Abou Ali, F. Dornaika, and J. Charafeddine, “Agentic AI: A Comprehensive Survey of Architectures, Applications, and Future Directions,” *Artificial Intelligence Review*, vol. 59, no. 1, Nov. 2025.

[9] Anthropic.”Model context protocol.” Internet: https://www.anthropic.com/news/model-context-protocol, Nov. 2024 [Nov. 5, 2025].

[10] Y. Yang, H. Chai, Y. Song, S. Qi, M. Wen, N. Li et al., “A Survey of AI Agent Protocols,” arXiv:2504.16736v3 [cs.AI], Jun. 2025.

[11] Z. Zhang, Q. Dai, X. Bo, C. Ma, R. Li, X. Chen et al., “A Survey on the Memory Mechanism of Large Language Model-based Agents,” *ACM Transactions on Information Systems*, vol. 43, no. 6, Sep. 2025.

[12] H. Derouiche, Z. Brahmi, and H. Mazeni, “Agentic AI Frameworks: Architectures, Protocols, and Design Challenges,” arXiv:2508.10146v1 [cs.AI], Aug. 2025

[13]T. Masterman, S. Besen, M. Sawtell, and A. Chao, “The Landscape of Emerging AI Agent Architectures for Reasoning, Planning, and Tool Calling: A Survey,” arXiv:2404.11584v1 [cs.AI], Apr. 2024.

[14] Z. Deng, Y. Guo, C. Han, W. Ma, J. Xiong, S. Wen, and Y. Xiang, “AI Agents Under Threat: A Survey of Key Security Challenges and Future Pathways,” *ACM Computing Surveys*, vol. 57, no. 7, p. 1–36, Feb. 2025.