International Journal of Latest Research in Engineering and Technology (IJLRET)
ISSN: 2454-5031
www.ijlret.com || Volume 11 - Issue 12 || December 2025 || PP. 63-68

Architectural Patterns of Multi Agent Systems for End to End Enterprise Workflows

Venkatesh Gundu
Senior Manager - Data Services & AI Platform
Columbus, Ohio, USA

Abstract: The study is devoted to the systematization and analytical examination of architectural patterns of multi-agent systems (MAS) aimed at automating end-to-end corporate business processes. The relevance is driven by a shift from monolithic AI models to flexible ensembles of specialized autonomous agents that jointly solve complex tasks, while the lack of established design practices hinders the construction of reliable and scalable solutions. The scientific novelty lies in proposing a taxonomy of MAS patterns — hierarchical, collaborative, and sequential — and in developing an original framework that enables the selection of an optimal pattern given the characteristics of the target business process. The work considers both classical and modern approaches to agent design and elucidates key aspects of their interaction: communication, coordination, and state management. Particular attention within the study is devoted to hybrid architectures as the most realistic option for complex corporate landscapes. The goal is to form a methodological foundation for MAS design in the corporate environment; to this end, pattern analysis, systems analysis, and conceptual modeling are employed. In conclusion, a decision-making framework is presented that helps architects align process requirements with the type of architecture. The material is intended for IT architects, engineers, and project managers in the field of AI automation.

Keywords: multi-agent systems, architectural patterns, corporate business processes, AI agents, LLM agents, automation, systems architecture, AutoGen, CrewAI, design pattern

I. Introduction

Modern organizations operate amid rapidly increasing, dynamic, and multidimensional business processes whose automation extends beyond the applicability of traditional robotic process automation (RPA) tools and isolated AI models. The current automation paradigm relies on multi-agent systems (MAS), ensembles of autonomous intelligent agents that coordinate interactions with one another and with the external environment for a unified goal. The emergence of large language models (LLM) as the cognitive core of such agents has sharply accelerated this shift. At the same time, industrial deployment of MAS is associated with serious architectural challenges: it is necessary to define mechanisms of inter-agent coordination, ensure reliability and controllability of the whole, and justify the choice of a structural solution for a specific task. The shortage of formalized architectural patterns leads to ad hoc constructions that are difficult to scale and maintain [2, 3]. This gap becomes particularly evident when analyzing the shortcomings of existing approaches. Traditional robotic process automation (RPA), being deterministic, exhibits brittleness under changes to the user interface or business logic and is unable to make autonomous decisions under uncertainty. On the other hand, monolithic AI models (for example, a single LLM), although strong at solving isolated tasks (NLU, CV, generation), do not have built-in capabilities for complex, multi-stage planning, coordination, and state management in a heterogeneous IT environment. They cannot autonomously execute an end-to-end process from start to finish, crossing the boundaries of different systems and departments.

This is precisely where the concept of the end-to-end enterprise business process comes into focus. In the context of this work, this is a multi-stage operation that crosses the boundaries of several functional departments (for example, from procure-to-pay to order-to-cash), requires the integration of heterogeneous IT systems (ERP, CRM, SCM), and often includes a human-in-the-loop for validation at key stages. The transition to an agentic paradigm is a direct response to this complexity: instead of a single all-knowing model, an ensemble of specialized agents is proposed, each of which is responsible for its own part of the process but coordinates its actions with others to achieve a common business goal.The **objective** of the work is to form a methodological framework for the analysis, classification, and well-founded selection of architectural patterns of multi-agent systems in the context of automating end-to-end corporate business processes.

Research tasks:

- Analyze and organize the existing spectrum of MAS architectural patterns, from classical models to modern LLM-oriented frameworks.

- Identify and classify key attributes of corporate processes, including the degree of determinism, complexity, and the need for human oversight, that influence the choice of architecture.
- Develop a conceptual decision framework designed to help systems architects select the best architectural pattern for a specific business process.

The scientific novelty lies in positioning the work as a kind of bridge between academic MAS research and corporate architecture practice. Unlike approaches focused on the behavior algorithms of individual agents, this study concentrates on the macro-level structures of their organization and interaction and proposes a practical tool for designing complex AI systems.

The author's hypothesis is based on the premise that the effectiveness of a multi-agent system in business process automation is determined not so much by the level of intelligence of individual agents as by the alignment of the chosen architectural pattern with the characteristics of the process itself. It is assumed that it is possible to formalize a framework that, based on process analysis, makes it possible a priori to select the optimal architecture, which significantly increases the likelihood of successful implementation.

II. Materials and Methods

The study relies on a critical examination of the corpus of scientific publications and on the analysis of technical documentation for current frameworks and their application in corporate environments.

Wu Q., Bansal G., Zhang J., Wu Y., Li B., Zhu E., & Wang C. [1] — establish a conversational coordination pattern for multi-agent LLM systems, where the roles of planner, executor, and critic exchange messages over a shared workspace and tools. This approach demonstrates how dialog protocols and explicit escalation to a human become an engineering-reproducible scaffold for end-to-end tasks.

Wang L., Ma C., Feng X., Zhang Z., Yang H., Zhang J., & Wen J. [2] — systematize the architectural building blocks of autonomous LLM agents (perception/tool-use, planning, multilayer memory, execution, self-assessment) and identify organizational interaction topologies (hierarchies, guilds, blackboard, market schemes), which provides a map of design choices for enterprise orchestration.

Bandi A., Kongari B., Naguru R., Pasnoor S., & Vilipala S. V. [3] — emphasize gaps in standards for assessing the reliability and safety of agentic AI in production environments.

Sumers T., Yao S., Narasimhan K., & Griffiths T. [5] — anchor LLM agents to cognitive architectures (attention, working/long-term memory, planning, metacognition), formalizing quality-control and reflection loops that can be encapsulated in the roles of evaluator and verifier of end-to-end business processes.

Movva H. [6] — shows the integration of agent orchestration with corporate RPA/orchestrators (using UiPath Maestro as an example), where task dispatching, capability catalogs, and execution telemetry form an observability contract between agents and the IT landscape; an emphasis on traceability and compliance makes the pattern practically applicable to E2E chains.

Panigrahy S. [7] — proposes a supra-system multi-agentic AI framework for enterprise digital transformation: agents discover automation opportunities, design microservices and LLM functions, and manage risks via an event bus, security policies, and solution quality agreements (SLO), which turns the agentic layer into a stratum above BPM/ESB.

Hsieh F.-S. [9] — formulates a context-aware workflow control scheme for cyber-physical systems, where a MAS dynamically reassigns process steps according to environmental state and real-time constraints; this context dispatcher pattern is directly transferable to production E2E scenarios with variable conditions.

Yao R., Hu Y., & Varga L. [4] — systematically review agent-based methods in multi-energy systems, highlighting stable architectural invariants: multilevel coordination (local–regional–global), market mechanisms for goal alignment, and built-in observability under strict SLA.

Izmirlioglu Y., Pham L., Son T. C., & Pontelli E. [10] — demonstrate how distributed dispatchers, coalitions, and hybrid hierarchies ensure decision coordination under network constraints; these domain references serve as field evidence of the viability of multi-agent coordination for complex, end-to-end, and time-sensitive processes.

Gronauer S., & Diepold K. [8] — provide the foundations of multi-agent deep reinforcement learning (MARL), covering centralized training with decentralized execution, credit assignment, robustness under partial observability, and the role of limited communication; such mechanisms are important for learning protocols of collaboration and role allocation under conflicting KPIs and uncertainty typical for enterprise pipelines.

However, gaps remain in the research, as some authors insist on rich metacontrol and agent self-assessment, whereas others demonstrate an industrial pull toward minimally sufficient roles and strict traceability. The least covered aspects are: benchmark task sets and unified metrics specifically for end-to-end corporate scenarios; formal contracts of correctness and security policies at agent–service interfaces; operationalization of reward shaping from business KPIs and compliance constraints; mechanisms for

determinization and audit of LLM-agent decisions under real-time requirements; as well as the legal and ethical attribution of responsibility in multi-agent solutions.

III. Results

Analysis of the literature and practical solutions shows that, when designing multi-agent systems (MAS) in a corporate environment, the determining factor is the chosen topology of inter-agent interaction. For comparability and reproducibility of solutions, it is advisable to standardize such topologies as architectural patterns and to link the choice of pattern to the formalized characteristics of the process being automated.

Taxonomically, most MAS reduce to three basic patterns and their hybrid combinations: hierarchical, collaborative, and sequential. This reduction does not negate the diversity of implementations but establishes a common descriptive language and facilitates design trade-offs among controllability, flexibility, and efficiency.

The hierarchical pattern (manager—executors) presupposes a central orchestrator agent that decomposes the objective function into sub-tasks, assigns them to specialized agents, and aggregates partial results. A representative example is a travel-planner agent that delegates searching for airline tickets, booking a hotel, and building an itinerary to profile executors. Key strengths are a transparent control flow, simplified debugging, and clear delineation of responsibility, while vulnerabilities include the risk of single points of failure and central node outage. In practice, the pattern is widespread, in particular in Lang Chain-based solutions with the Agent Executor component. The key difficulty of this pattern lies not only in the risk of creating the most vulnerable point and a single point of failure (SPOF) in the central orchestrating agent, but also in the complexity of state management. The manager must not only delegate, but also track the lifecycle of subtasks, handle exceptions and timeouts, which requires the implementation of fault-tolerant supervisors. In industrial implementations this often requires the use of asynchronous task queues to decouple the manager and the executors, which, in turn, complicates end-to-end process tracing.

The collaborative pattern (round table/swarm) treats agents as peers who coordinate actions directly (peer-to-peer) or through a shared communication space to jointly develop a solution. A characteristic scenario is the joint work of a Data Analyst, a Domain Expert, and a Visualizer in a shared chat on a single dataset with hypothesis generation and verification. Advantages include high adaptability and suitability for weakly formalized, creative tasks; limitations include complicated coordination, the risk of discussion loops, and unpredictability of outputs. A representative implementation is the agent conversation in Microsoft AutoGen. Technically, this pattern is implemented via a blackboard or shared context to which agents contribute their proposals. The fundamental problem is to ensure convergence. Without clear moderation mechanisms (for example, a dedicated chat manager agent or voting rules), agents risk entering discussion loops or mutually reinforcing hallucinations. Furthermore, the high degree of iterativity makes this pattern computationally and financially expensive, which requires taking token economics into account at the design stage.

The sequential pattern (assembly line) organizes the process as a chain of stages with a strictly defined order of artifact passage: each agent performs a strictly specified operation and passes the result further. For example, a News Collector extracts materials, a Summarizer condenses content, a Translator localizes it, and a Publisher posts it to a blog. This approach ensures determinism, predictability, and high efficiency for linear pipelines but is characterized by low flexibility and high sensitivity to errors at any stage. Modern frameworks such as CrewAI provide direct support for such sequential scenarios [1, 4]. The foundation of the pattern is a strict data contract, an artifact passed between agents. The vulnerability of this approach is the cascading propagation of an error. A failure at a single stage (for example, the translator agent) stops the entire chain. Therefore, for production readiness, this pattern requires the immediate implementation of exception handling mechanisms for each Task, dead-letter queues for failed artifacts and, in complex cases, compensating transaction patterns to roll back previous steps.

The choice among these patterns is not arbitrary and must be derived from the characteristics of the business process itself. The key dimensions are determinism, complexity, and the required level of autonomy. Determinism reflects the predictability and regulation of procedures (high — invoice processing; low — development of a marketing strategy). Complexity characterizes the number of interrelated factors and the need for interdisciplinary expertise (low — classification of incoming mail; high — analysis of causes of failures in supply chains). The level of autonomy sets the permissible depth of automation without human involvement (low — mandatory confirmation of each step; high — end-to-end automation turnkey).

These dimensions directly determine the architectural choice. Highly deterministic, linear processes with requirements for stable throughput and quality control are optimal for the sequential pattern. Tasks with low determinism and high domain complexity benefit from collaborative configurations that enable emergent solutions through iterative exchange. Hierarchical structures are appropriate for moderate complexity and the need for managerial discipline, centralized responsibility, and SLA compliance; they should be complemented with mechanisms for fault tolerance and offloading of the central node. In real corporate scenarios, hybrids are

often applied: for example, a collaborative round table for hypothesis generation is embedded as a stage in an overall sequential pipeline, and final integration of results is assigned to a hierarchical manager [1, 3]. This compositional approach aligns task variability with the required controllability of the system, minimizing the risks and costs of switching patterns as processes evolve. Thus, relying solely on patterns is rather an academic idealization. The main result of the analysis is that hybrid compositions are the dominant solution in corporate environments. For example, the Hierarchical-Sequential pattern (for controlled decomposition into deterministic sub-chains, as in CrewAI integrated into Lang Chain) or the Hierarchical-Collaborative pattern (for encapsulation of a non-trivial creative step, for example, a brainstorming phase by a group of AutoGen agents within an overall managed process).

IV. Discussion

The summary conclusions of the analysis confirm that a universal methodology, that is, a single unconditionally best architectural pattern, does not exist. The choice must be deliberate and context-dependent, based on a deep, domain-specific understanding of the target business process. To formalize such a choice, the Enterprise Agent Architecture Selection Framework (EAAS) is proposed.

EAAS includes three interrelated components: a process classification tool, a pattern correspondence map, and a decision matrix. The first component, Business Process Characteristics Cube, defines a parametric space in which any automation task can be correctly visualized and unambiguously positioned (figure 1).

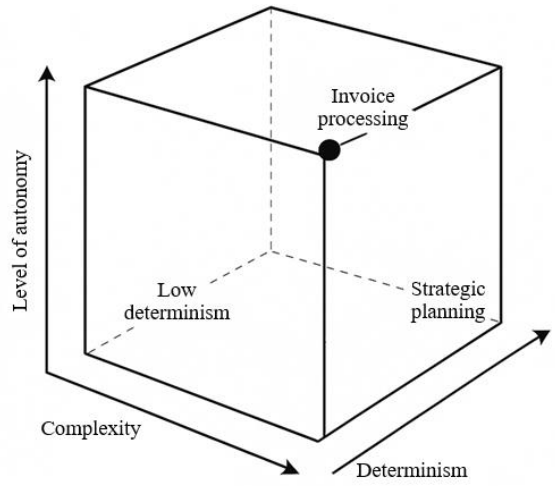

Figure 1: Cube of business process characteristics [3, 5, 9]

Having established the classification of the process in the Business Process Characteristics Cube, the architect consults the Pattern Correspondence Map for preliminary selection (figure 2).

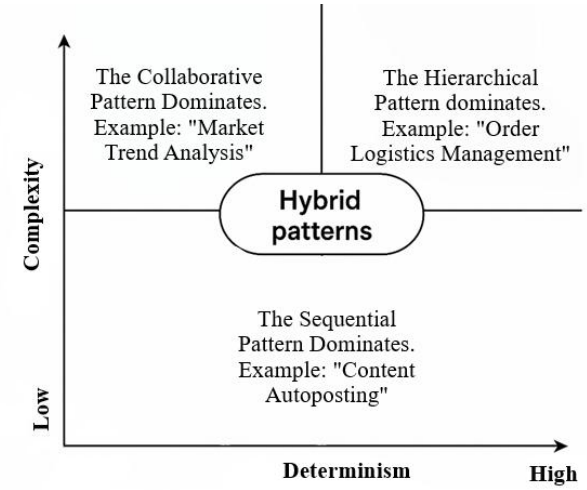

Figure 2: Matching map of patterns and types of processes [3, 7]

In practice, the corporate environment is characterized not by pure architectural solutions but by their hybrid compositions. As an illustration, consider automating the process of launching a new promotional campaign in the retail segment.

Campaign Agent-Manager (hierarchy) receives a task and initiates three subprocesses.

Research Group (collaboration): agents Market Analyst, SEO Specialist, and Competitor Analyst communicate in a shared chat and jointly develop the campaign concept.

Content Factory (sequence): the Copywriter writes the text → the Designer creates the visuals → the Email Markup Specialist assembles the email.

Logistics Coordinator (hierarchy/task): the Inventory Check agent verifies product availability in the warehouse.

The Campaign Manager aggregates the results of all subprocesses and transfers them to the Human Marketer for final approval.

The example clearly demonstrates how different patterns are combined to automate a single end-to-end business process. For further refinement and justification of the selection, the EAAS Decision Matrix is employed [6, 7] (table 1).

Table 1: EAAS Decision Matrix [3, 6, 7]

Process characteristic	Sequential	Hierarchical	Collaborative
Determinism	(Ideal)	Medium	Low alignment
Complexity (multifactor)	Low alignment	Medium	High
Creativity required	Low alignment	Low alignment	High
Outcome predictability	High	Medium	Low alignment
Ease of debugging	High	Medium	Low alignment

Ultimately, the selected pattern predetermines the configuration and boundaries of the technology stack (table 2).

Table 2: Mapping of frameworks and architectural patterns [7, 8, 10]

Framework	Primary supported pattern	Flexibility	Entry barrier
Lang Chain (Agents)	Hierarchical, Sequential	High	Medium
Microsoft AutoGen	Collaborative	Medium	High
CrewAI	Sequential, Hierarchical	Low	Low

It must be emphasized that the proposed EAAS framework carries not only technical but also organizational and economic consequences. The choice of pattern, guided by the framework, directly impacts the total cost of ownership (TCO) and the solution's governance model.

– Economics (Tokenomics): Collaborative patterns, requiring dozens of LLM iterations to reach consensus, are expensive in terms of "token cost" and response time. Sequential patterns are "cheap" to execute but "expensive" to design due to the need to formalize rigid data contracts. EAAS makes this trade-off deliberate.

– Audit and Governance: The patterns dictate the audit model. Sequential processes are easy to trace (step-by-step log). Hierarchical ones require end-to-end tracing of the "task_ID." Collaborative patterns, due to their stochastic nature, present the greatest difficulty for auditing and compliance, requiring logging of the entire "chat" for post-analysis.

For a thorough academic analysis, the limitations of the proposed EAAS must be acknowledged. First, in its current form, it is a qualitative tool. The axes of the "Characteristics Cube" (Fig. 1), such as "complexity" and "determinism," lack strict quantification, leaving room for the architect's subjective assessment. Second, the framework does not explicitly include Non-Functional Requirements (NFRs), such as "latency" or "throughput." For example, the collaborative pattern is inherently slow, and EAAS in its current form does not indicate scenarios where this is unacceptable.

Thus, the proposed EAAS framework provides systems architects with a rigorous methodological foundation for addressing one of the most challenging contemporary problems — the design of multi-agent systems. Instead of intuitive choices dictated by the popularity of tools, it ensures a reasoned selection of solutions derived from the specifics of the business process being automated, which reduces risks and increases the likelihood of successful project implementation.

V. Conclusion

The study achieved its stated goal — a methodological framework for selecting MAS architectural patterns for the enterprise environment has been created. All research tasks were addressed sequentially: a systematization was conducted and a rigorous description provided of the three basic schemes — sequential, hierarchical, and collaborative — with emphasis on the significance of their hybrid combinations; the properties of business processes that determine the choice of architecture were identified, namely the degree of determinism, the level of complexity, and the level of autonomy; the EAAS framework was developed, comprising the Characteristics Cube, the Alignment Map, and the Decision Matrix, serving as an applied tool for architects of AI systems. Empirical and analytical results confirmed the working hypothesis: a universally best architecture does not exist, the effectiveness of MAS is conditioned by aligning the selected pattern with the characteristics of the process being automated, and EAAS formalizes and makes the very act of selection reproducible. Thus, as the era of agentic AI moves from R&D to industrial operation, intelligence becomes a resource (commodity), and architecture becomes the lever of its application. The key conclusion of this work is that it is methodological rigor in interaction design, rather than the mere intelligence of individual agents, that becomes the main differentiator between controlled, reliable corporate systems and fragile, unmaintainable prototypes. The EAAS framework offers a first step toward such engineering, shifting the focus from programming agents to designing their societies.

References

[1]. Wu, Q., Bansal, G., Zhang, J., Wu, Y., Li, B., Zhu, E., & Wang, C. (2024, August). Autogen: Enabling next-gen LLM applications via multi-agent conversations. In First Conference on Language Modeling, 1-46.

[2]. Wang, L., Ma, C., Feng, X., Zhang, Z., Yang, H., Zhang, J., & Wen, J. (2024). A survey on large language model based autonomous agents. Frontiers of Computer Science, 18(6), 186345. https://doi.org/10.1007/s11704-024-40231-1

[3]. Bandi, A., Kongari, B., Naguru, R., Pasnoor, S., & Vilipala, S. V. (2025). The Rise of Agentic AI: A Review of Definitions, Frameworks, Architectures, Applications, Evaluation Metrics, and Challenges. Future Internet, 17(9), 404. https://doi.org/10.3390/fi17090404

[4]. Yao, R., Hu, Y., & Varga, L. (2023). Applications of Agent-Based Methods in Multi-Energy Systems—A Systematic Literature Review. Energies, 16(5), 2456. https://doi.org/10.3390/en16052456

[5]. Sumers, T., Yao, S., Narasimhan, K., & Griffiths, T. (2024). Cognitive architectures for language agents. Transactions on Machine Learning Research, 1-32.

[6]. Movva, H. (2025). Building AI Agents and Orchestration using UiPath Maestro. International Journal of Artificial Intelligence, Data Science, and Machine Learning, 6(3), 32-37. https://ijaidsml.org/index.php/ijaidsml/article/view/234

[7]. Panigrahy, S. (2025). Multi-Agentic AI Systems: A Comprehensive Framework for Enterprise Digital Transformation. Journal of Computer Science and Technology Studies, 7(6), 86-96.

[8]. Gronauer, S., & Diepold, K. (2022). Multi-agent deep reinforcement learning: a survey. Artificial Intelligence Review, 55, 895-943.

[9]. Hsieh, F.-S. (2021). A Dynamic Context-Aware Workflow Management Scheme for Cyber-Physical Systems Based on Multi-Agent System Architecture. Applied Sciences, 11(5), 2030. https://doi.org/10.3390/app11052030

[10]. Izmirlioglu, Y., Pham, L., Son, T. C., & Pontelli, E. (2024). A Survey of Multi-Agent Systems for Smartgrids. Energies, 17(15), 3620. https://doi.org/10.3390/en17153620