Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Agentic AI: A Comprehensive Survey of Foundations, Architectural Paradigms, Core Capabilities, and a Unified Reference Architecture

Adabara Ibrahim

ibrahimadabara@kiu.ac.ug

Kampala International University

Abigaba Sofia

Kampala International University

Kyampaire Sarah

Kampala International University

Rumanzi Smith

Kampala International University

Kisembo Moreen

Kampala International University

Yale Ibrahim Danjuma

Kampala International University

Elhadi Abdalla A. Arbab

Kampala International University

Venkateswarlu Maninti

Kampala International University

Kanyesigye Simon

Kampala International University

Ninyesiga Allan

Kampala International University

Research Article

Keywords: Agentic AI, intelligent agents, agent architectures, architecture-capability mapping, neuro-symbolic AI, multi-agent systems

Posted Date: November 18th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11461890/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Agentic AI: A Comprehensive Survey of Foundations, Architectural Paradigms, Core Capabilities, and a Unified Reference Architecture

Ibrahim Adabara[1], **Abigaba Sofia**[1], **Kyampaire Sarah**[1], **Rumanzi Smith**[1], **Kisembo Moreen**[1], **Yale Ibrahim Danjuma**[1], **Elhadi Abdalla A. Arbab**[1], **Venkateswarlu Maninti**[1], **Kanyesigye Simon**[1], **Ninyesiga Allan**[1]

[1]Department of Computing, Faculty of Science and Technology, Kampala International University, Western Campus, Bushenyi, Western Region, Uganda

Corresponding author: Ibrahim Adabara | ibrahimadabara@kiu.ac.ug | ORCID: 0000-0001-8949-7540

Abstract

Agentic artificial intelligence (Agentic AI) integrates goal-directed decision-making, state continuity, planning, tool-mediated action, and feedback-driven adaptation into interactive computational systems. Although existing surveys examine agent capabilities, architectures, and applications, cross-paradigm comparisons often conflate functional capabilities with implementation mechanisms and organizational structures. This comprehensive narrative survey synthesizes foundational and contemporary research on symbolic/classical, neural/generative, neuro-symbolic, and multi-agent approaches to clarify conceptual boundaries and examine how agentic capabilities are architecturally realized. It organizes recurring functions into nine groups: goal and context management, perception and state representation, reasoning and decision-making, planning and replanning, memory and knowledge, action and tool use, evaluation and reflection, learning and adaptation, and coordination and orchestration. The principal contribution is an architecture-capability mapping separating functional responsibilities from computational mechanisms and coordination structures. A complementary conceptual reference architecture integrates goal management, environmental interaction, planning, memory, execution, evaluation, and cross-cutting assurance concerns. Representative cases demonstrate how different systems can realize similar functions using different mechanisms. The proposed taxonomy, formal abstraction, and reference architecture are analytical synthesis artifacts rather than experimentally validated models. They support transparent architectural comparison and identify testable research questions.

Keywords: Agentic AI; intelligent agents; agent architectures; architecture-capability mapping; neuro-symbolic AI; multi-agent systems.

1. Introduction

1.1 Motivation and Background

Artificial intelligence has progressively moved from systems designed for narrowly specified tasks toward systems capable of more persistent, interactive, and goal-directed behavior. Early symbolic AI emphasized explicit representation, search, problem solving, and planning, establishing computational foundations for goal-oriented action (Newell & Simon, 1976). Intelligent-agent research subsequently formalized properties such as autonomy, reactivity, proactivity, and social interaction (Wooldridge & Jennings, 1995), while cognitive architectures sought to integrate knowledge, goals, reasoning, memory, learning, and action within persistent computational systems (Laird et al., 1987). Reinforcement learning provided a complementary perspective in which agents improve behavior through repeated interaction with an environment and feedback from their actions (Mnih et al., 2015).

The emergence of transformer-based foundation models significantly expanded the representational and generative capabilities available to intelligent systems. Transformers enabled scalable contextual sequence processing (Vaswani et al., 2017), while large language models demonstrated increasingly general language understanding, generation, and few-shot adaptation (Brown et al., 2020). Foundation models subsequently provided reusable computational substrates that could be adapted across a wide range of tasks (Bommasani et al., 2021). These developments established the technological basis for Generative AI, whose primary emphasis is the production of new content conditioned on user or environmental input (Feuerriegel et al., 2024).

Contemporary AI agents extend this generative capability by coupling foundation models with mechanisms for reasoning, planning, memory, external action, and environmental feedback. ReAct integrates reasoning with iterative action and observation (Yao et al., 2023a); Toolformer demonstrates interaction between language models and external tools (Schick et al., 2023); Generative Agents combine memory, reflection, and planning to support persistent behavior (Park et al., 2023); and conversational multi-agent frameworks extend such capabilities across interacting specialized agents (Wu et al., 2023). These developments shift the unit of analysis

from an isolated model response toward an integrated system capable of sustaining goal-directed behavior over multiple interactions.

Agentic Artificial Intelligence has emerged within this convergence. Recent literature associates Agentic AI with increasing autonomy, multi-step goal pursuit, contextual reasoning, planning, persistent or external memory, tool-mediated action, feedback, adaptation, and, in some systems, multi-agent coordination (Hughes et al., 2025; Schneider, 2025). Its intellectual foundations therefore extend beyond the current generation of LLM agents. Contemporary Agentic AI inherits concepts and mechanisms from classical planning, cognitive architectures, reinforcement learning, autonomous-agent research, multi-agent systems, and foundation-model-based agents.

This convergence suggests that Agentic AI should not be interpreted simply as the next stage of language-model scaling. It represents an architectural problem concerning how goals, state, reasoning, planning, memory, action, feedback, adaptation, and coordination are assembled into systems capable of operating with reduced dependence on continuous human specification.

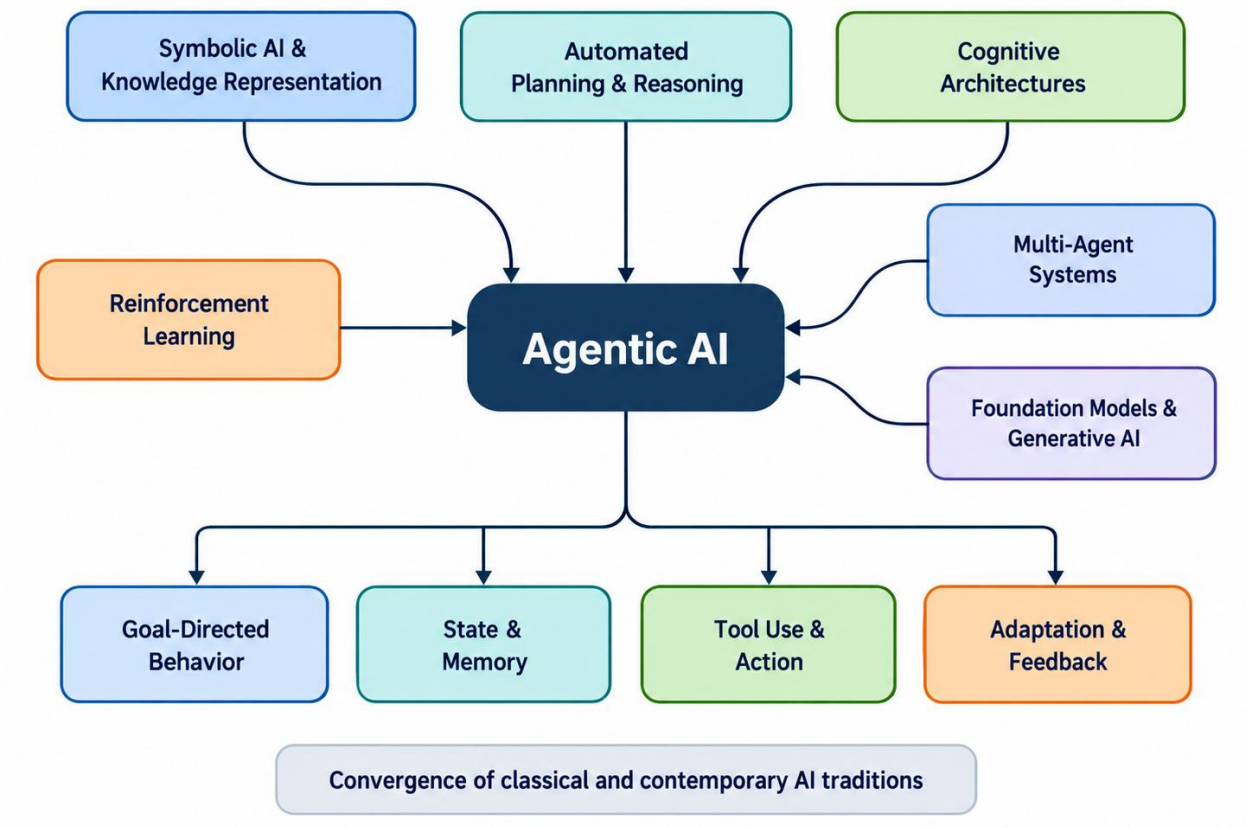


Fig. 1. Intellectual foundations, convergent architectural traditions, and conceptual positioning of Agentic AI. Conceptual visualization prepared with generative-AI assistance; design and scientific content require author verification.

1.2 Definition and Scope

Despite growing research activity, Agentic AI does not yet have a universally stabilized definition. The literature uses related terms including AI agent, autonomous agent, LLM agent, agentic system, multi-agent system, and Agentic AI, sometimes at different levels of abstraction. This creates ambiguity concerning whether Agentic AI refers to an individual agent, a collection of interacting agents, an LLM-based software architecture, or a broader paradigm of autonomous intelligent systems.

Recent conceptual analyses illustrate this variation. Sapkota et al. (2026) distinguish AI agents from the broader concept of Agentic AI, while Bansod (2025) differentiates individual autonomous agents from collaborative agentic systems. Botti (2025) further questions how contemporary Agentic AI relates to the established multi-agent systems tradition. These distinctions indicate that the central problem is not simply terminological. Different definitions imply different assumptions concerning the unit of agency and the capabilities required for a system to qualify as agentic.

This survey therefore treats Agentic AI as an architectural paradigm and systems concept rather than as a synonym for an LLM agent or a particular model family. Under this perspective, foundation models, planners, retrieval systems, memory stores, tools, and individual agents are potential constituents of an Agentic AI system, but none alone is sufficient to define the paradigm.

The primary scope of the survey is the relationship among the foundations, core capabilities, architectural paradigms, and realization mechanisms of Agentic AI. Particular attention is given to perception, goal management, reasoning, planning, memory, tool use, reflection, adaptation, and coordination. Explainability and Responsible AI are treated as cross-cutting architectural considerations because autonomy creates requirements for traceability, oversight, safety, accountability, privacy, and security.

The survey does not aim to provide an exhaustive catalogue of domain applications, regulations, benchmark suites, or socioeconomic effects. Application cases are used selectively when they clarify an architectural mechanism or capability.

1.3 Research Problem, Knowledge Gaps, and Novelty

A central difficulty in the study of Agentic AI is not the absence of surveys but the coexistence of several legitimate units of analysis. Wang et al. (2024) and Masterman et al. (2024) address foundation-model agents and agent architectures, while Acharya et al. (2025) and Bandi et al. (2025) synthesize broader definitions, capabilities, applications, and challenges. Abou Ali et al. (2026) explicitly separate symbolic/classical and neural/generative lineages; Peykani et al. (2026a) discuss frameworks, infrastructure, evaluation, and safety. These contributions establish substantial prior art and preclude claiming that an architectural survey, taxonomy, or reference architecture is itself unprecedented.

Recent comparisons create a more demanding novelty test. Arunkumar et al. (2026) propose a unified taxonomy of LLM-based agents comprising perception, cognition, planning, action, tool use, and collaboration. Khapre et al. (2026) combine architecture, cognitive foundations, interaction, trust, and evaluation in a multidimensional systematic survey. Hakim et al. (2026) analyze neuro-symbolic integration patterns, while Bajoria et al. (2026) propose a modular LLM-agent reference architecture with autonomy, tool-use, collaboration, and governance axes. Accordingly, the present article does not assert priority for a unified taxonomy or modular agent reference architecture. Its narrower proposed contribution is a cross-paradigm account of function-to-mechanism realization that does not assume an LLM-centric implementation.

The key analytical distinction is between a capability label and its realization mechanism. Planning may mean formal state-space search, LLM-generated decomposition, constrained neuro-symbolic optimization, or distributed subtask assignment. Memory may refer to symbolic beliefs, context windows, episodic stores, retrieval indexes, procedural skills, or shared state. Such systems can perform functionally similar tasks without being architecturally equivalent. Treating multi-agent composition as an organizational dimension, rather than an exclusive peer to symbolic and neural implementations, further avoids conflating control topology with computational mechanism.

The specific research question is therefore how to describe and compare these many-to-many relationships consistently. Sections 3–6 distinguish a functional capability layer, realization mechanisms, and emergent system properties, then map concrete mechanisms to the reference architecture. This is a proposed analytical synthesis, not a proof of novelty against every publication or an empirically validated architecture. Its value must be judged by transparent boundaries, traceability to representative studies, and whether the mapping exposes design choices that competing frameworks otherwise make difficult to compare.

Table 1. Positioning of the present survey relative to representative prior reviews.

Representative study	Primary emphasis	Distinctive focus of this survey
Wang et al. (2024)	LLM-based autonomous agents	Covers symbolic and hybrid realizations alongside neural agents
Acharya et al. (2025)	Broad foundations, methodologies and applications	Centers capability-to-mechanism architectural relationships
Abou Ali et al. (2026)	Symbolic/classical and neural/generative lineages	Maps shared functions to distinct realization mechanisms
Gohil et al. (2026)	Foundations, trustworthiness and applications	Treats assurance as an architectural boundary
Nageshwaran et al. (2026)	Agent evaluation meta-taxonomy	Distinguishes evaluation evidence from mechanism descriptions
Ala'anzy et al. (2026)	Modular agent architecture and MAS survey	Separates computational realization from organizational topology

The novelty argument must be bounded in relation to surveys that already classify agent architectures, integrate modular components or propose reference models. The contribution here is not priority for a taxonomy as such: it is an evidence-grounded distinction among functions, computational realization mechanisms and organizational topology. A function-to-mechanism

mapping is most useful when it records information dependencies, action interfaces, assurance requirements and the source of each coding decision.

A further challenge is conflation of classification dimensions. Symbolic, neural and neuro-symbolic identify computational approaches, whereas multi-agent, hierarchical and workflow-based describe arrangements of responsibility and execution. A taxonomy that treats them as interchangeable categories can hide the fact that a multi-agent system may itself contain heterogeneous symbolic and neural components. The present survey therefore separates these dimensions and proposes an explicit coding instance for each capability-realization relationship.

1.4 Research Objectives and Contributions

The objective is to develop and articulate an architecture-centered, paradigm-neutral analytical account of Agentic AI. Six contributions are proposed. C1 specifies an operational system-level definition that distinguishes Agentic AI from related constructs, with acknowledged overlap. C2 organizes recurrent functions into nine capability groups. C3 distinguishes symbolic/classical, neural/generative, hybrid neuro-symbolic, and multi-agent composition as analytical lenses while recognizing that the last is also a system organization choice. C4 provides a capability-to-realization mapping showing multiple ways to implement a common function. C5 proposes an integrated taxonomy and reference architecture, together with an illustrative formal abstraction of recurrent interaction. C6 identifies explainability, human oversight, safety, accountability, privacy, and security as cross-cutting design obligations. These are literature-grounded conceptual outputs; neither architectural superiority nor deployment efficacy is experimentally established by this survey.

1.5 Research Questions

The survey is guided by four interdependent research questions. RQ1.1 asks how Agentic Artificial Intelligence is conceptualized and distinguished from related constructs such as AI agents, Generative AI, autonomous systems, and multi-agent systems. RQ1.2 asks which architectural components and core capabilities are consistently associated with Agentic AI systems. RQ1.3 asks how symbolic/classical, neural/generative, hybrid, and multi-agent architectures differ in the mechanisms through which they realize agentic capabilities. RQ1.4 asks how the relationships among perception, reasoning, planning, memory, reflection, tool use,

adaptation, and coordination can be synthesized into an integrated architecture-capability taxonomy for Agentic AI.

Table 2. Research questions and corresponding scholarly outputs.

Research question	Analytical focus	Expected output
RQ1	Definitions and conceptual boundaries	Operational definition of Agentic AI
RQ2	Capabilities and emergent properties	Nine-group capability taxonomy
RQ3	Computational mechanisms and organization	Cross-paradigm realization matrix
RQ4	Integrated architecture relationships	Taxonomy, coding schema and reference architecture

1.6 Survey Methodology and Analytical Synthesis Approach

This work is a comprehensive, interpretive survey and conceptual synthesis, not a systematic review, scoping review, or meta-analysis. Its source-gathering description covers the ACM Digital Library, IEEE Xplore, Scopus, Web of Science, SpringerLink, ScienceDirect, and arXiv, using combinations of terms for agentic AI, intelligent or autonomous agents, agent architecture, reasoning, planning, memory, reflection, tool use, orchestration, and multi-agent systems. Its purpose is to identify historically relevant agent principles alongside contemporary mechanisms and reviews, rather than estimate publication frequencies or pooled effects.

The synthesis prioritizes foundational primary works on agency and representative research that specifies a definable capability, implementation mechanism, coordination structure, or architectural trade-off. Review articles are used to establish the competitive landscape; preprints remain explicitly distinguishable from peer-reviewed publications. Backward and forward citation tracing were reported in the initial manuscript. The corpus incorporates works available through October 2026. No preregistered protocol, exhaustive record-by-record screening log, appraisal scores, or reproducible database export was supplied with this draft; therefore, completeness, screening counts, and quantitative prevalence are not claimed. Analytical coding separates (i) the function performed, (ii) the implementation mechanism, (iii) the host

architectural paradigm or organization, (iv) state/action interfaces, and (v) limitations and assurance dependencies. The resulting comparisons are interpretive and should not be read as effect estimates or experimentally demonstrated rankings.

The analysis is a comprehensive narrative synthesis, not a systematic review or meta-analysis. The reported searches and citation tracing were targeted to foundational, representative and recent architectural work; no exhaustive screening register, preregistered protocol, risk-of-bias score, independent coder assessment or database-specific study-count series is available. The absence of these materials prevents prevalence estimates and claims of exhaustive coverage.

Five fields are used to interpret each representative architectural example: the function being realized; its computational mechanism; the organizational context in which it operates; the state and action interfaces it requires; and the assumptions, limitations and assurance controls that condition its use. Evidence that is directly documented is distinguished from a reasoned interpretive assignment. Where information is not reported, the corresponding field is treated as uncertain rather than coded as absent.

1.7 Organization of the Paper

Section 2 clarifies definitions and boundaries; Section 3 identifies functional capabilities; Section 4 describes architectural paradigms and coordination structures; Section 5 compares mechanisms and trade-offs; and Section 6 integrates these observations into a taxonomy, mapping, formal abstraction, and proposed reference architecture. Section 7 discusses cross-cutting assurance concerns, Section 8 technical failure modes, and Section 9 research directions. Section 10 states the survey limitations, followed by the conclusion and disclosures.

2. Foundations and Conceptual Boundaries of Agentic AI

Agentic AI draws on several older traditions in artificial intelligence, yet its contemporary form combines them in ways that make direct equivalence difficult. Classical planning, cognitive architectures, reinforcement learning, autonomous-agent theory, multi-agent systems, and foundation-model-based agents all contribute concepts that now appear in Agentic AI. This section addresses RQ1.1 by clarifying this lineage and establishing the conceptual boundaries required for the remainder of the survey.

2.1 Intellectual Foundations of Agency in AI

The foundations of agency in AI predate contemporary foundation models. Early planning systems represented intelligent action through explicit states, goals, preconditions, and effects. STRIPS established a formal basis for constructing action sequences toward desired states (Fikes & Nilsson, 1971). These ideas established planning, state transition, and goal satisfaction as enduring elements of agent architecture.

Agent research subsequently shifted attention from individual algorithms toward persistent computational entities situated in environments. BDI architectures formalized deliberation through beliefs, desires, and intentions, while later surveys show how this framework supports goal commitment, reasoning, and action selection (de Silva et al., 2020). Cognitive architectures extended these systems view by integrating memory, perception, reasoning, and action. ACT-R represents one influential approach to modeling declarative and procedural cognition within a unified architecture (Anderson et al., 1997).

Reinforcement learning introduced another account of agency based on repeated interaction, feedback, and behavioral adaptation. Partial observability further formalized decision-making where agents must act without complete knowledge of environmental state (Kaelbling et al., 1998). Multi-agent research broadened the unit of intelligence from an individual agent to interacting populations. Communication, cooperation, competition, coordination, and distributed problem solving became architectural concerns in their own right (Jennings et al., 1998; Stone & Veloso, 2000).

The contemporary foundation-model era adds flexible language understanding, broad pretrained knowledge, natural-language reasoning, and general-purpose tool interaction. Recent architecture surveys characterize LLM-based agents as systems in which foundation models function as reasoning components within wider structures containing planning, memory, tools, and environmental interaction (Durante et al., 2024; Xi et al., 2025). The conceptual lineage is therefore better understood as convergence rather than replacement.

2.2 Agentic AI in the Contemporary AI Ecosystem

Agentic AI occupies a systems-level position within the contemporary AI ecosystem. It overlaps with Generative AI, autonomous agents, and multi-agent systems but is not reducible to any one of them. Raghavendra and Saikia (2026) distinguish an AI agent, an agentic AI system, and

Agentic AI as a broader paradigm. Under this formulation, an individual agent perceives, reasons, and acts, while an agentic AI system integrates one or more such agents with memory, planning, tools, and orchestration.

Accordingly, this survey treats Agentic AI as a systems and architectural concept. An LLM may serve as the reasoning core, but the complete system may also require state, memory, planning, execution, feedback, and coordination.

2.3 Agentic AI versus Traditional AI

Many task-specific AI applications are optimized for a bounded function such as classification, prediction, recommendation, recognition, or control. Its objective and operational boundaries are largely specified before execution. Agentic AI differs primarily in how tasks are organized over time. It can interpret a higher-level objective, decompose it, maintain state, select actions, interact with external resources, and adapt subsequent behavior according to results (Sujatha, 2025).

2.4 Agentic AI versus Generative AI

Generative AI and Agentic AI are closely related but describe different functions. Generative AI centers on producing new content from learned representations. Agentic AI centers on pursuing objectives through iterative action. A generative model can produce text, code, images, or plans without executing them, monitoring their consequences, or maintaining task state. Agentic systems connect generation to memory, planning, tools, execution, and feedback.

2.5 AI Agents versus Agentic AI Systems

An AI agent can be understood as an individual computational entity that perceives an environment, reasons about available information, and acts toward an objective. An agentic AI system operates at a broader architectural level by integrating one or more agents with supporting mechanisms such as planning, memory, tools, feedback, and orchestration (Raghavendra & Saikia, 2026). Recent architecture-oriented work reinforces this distinction by treating AI agent systems as compound architectures rather than isolated reasoning models (Xu, 2026; Derouiche et al., 2025).

An AI agent is an acting entity. An agentic AI system is the operational architecture that enables sustained agency.

2.6 Agentic AI versus Autonomous Intelligent Systems

Autonomy is necessary to understand Agentic AI, but the concepts are not identical. Autonomous intelligent systems include a broad range of software and embodied systems capable of functioning with limited direct human intervention. Agentic AI is more specific because autonomy is organized around goal-directed cognitive and executive mechanisms. Recent work on autonomy levels for AI agents emphasizes that systems vary in how much decision authority they exercise rather than belonging to a simple autonomous or non-autonomous binary (Feng et al., 2025).

2.7 Agentic AI and Multi-Agent Systems

Multi-agent systems and Agentic AI share a strong conceptual overlap because both can involve autonomous entities coordinating within a shared environment. Traditional MAS research, however, focuses specifically on interaction among multiple agents, while Agentic AI may also be realized through a sufficiently capable single-agent architecture. The recent literature does not support a universal requirement that Agentic AI must contain multiple agents. Multi-agent organization is therefore better treated as a coordination structure that may contain symbolic, neural, or hybrid agents.

Table 3. Conceptual boundaries between Agentic AI and adjacent AI constructs.

Construct	Unit of analysis	Defining emphasis	Relationship to Agentic AI
Task-specific AI	Model or bounded application	Prediction or predefined function	Potential constituent capability
Generative AI	Generative model/application	Learned content generation	Possible reasoning or planning mechanism
AI agent	Computational acting entity	Situated decisions and actions	May constitute an agentic system
Autonomous system	Controller or operational system	Independent execution within constraints	Substantial conceptual overlap
Multi-agent system	Multiple interacting agents	Coordination and distributed behavior	Optional organizational configuration

Construct	Unit of analysis	Defining emphasis	Relationship to Agentic AI
Agentic AI system	Integrated operational architecture	Stateful and feedback-sensitive goal pursuit	Focal unit of this survey

2.8 Definitions of Agentic AI Across the Literature

Definitions of Agentic AI vary mainly along three dimensions: the unit of agency, the minimum required capabilities, and the importance assigned to multi-agent coordination. Existing definitions converge around goal-directed autonomy and environmental interaction, but differ on whether reflection, persistent memory, learning, or multiple agents are mandatory.

Table 4. Representative definitional and architectural perspectives in the literature.

Study	Conceptual emphasis	Unit of analysis	Relevance
Acharya et al. (2025)	Autonomous goal pursuit	Agentic system	Broad operational properties
Bandi et al. (2025)	Planning, memory and tool use	Agentic architectures	Recurring functional capabilities
Abou Ali et al. (2026)	Symbolic and neural lineages	Architecture families	Distinct realization mechanisms
Raghavendra & Saikia (2026)	Agent versus system	Integrated system	Unit-of-analysis clarity
Li et al. (2026)	Single- and multi-agent forms	System organization	Topology distinction
Gohil et al. (2026)	Foundations and trustworthy operation	Agentic systems	Assurance implications

2.9 Proposed Operational Definition of Agentic AI

The evidence supports several conclusions. Agentic AI should be defined at the system level, even when only one agent is present. Goal-directed autonomy is central but insufficient on its own. Agentic behavior requires recurrent interaction in which environmental information

influences subsequent decisions. Some form of state or memory continuity is needed for sustained multi-step behavior. Planning, reasoning, and action must be functionally connected, while multi-agent coordination can extend agenticity without being mandatory.

Definition 1. Agentic Artificial Intelligence

Agentic Artificial Intelligence refers to a systems paradigm in which one or more computational agents pursue goals through recurrent perception or state acquisition, reasoning and decision-making, planning or action organization, environmental interaction, and feedback, supported by sufficient state or memory continuity to sustain multi-step behavior with reduced need for continuous human specification.

The definition does not require persistent parameter learning, unrestricted autonomy, multiple interacting agents or the presence of a large language model. The operational minimum is a recurrent process that maintains sufficient state to relate objectives, decisions and actions to subsequent environmental feedback. A classical agent satisfying these conditions falls within scope, even when it predates contemporary terminology.

Functional responsibilities should also be distinguished from system-level characteristics. Autonomy, adaptable behavior and long-horizon persistence cannot be inferred solely from labels attached to modules. Their presence depends on the integration of goals, state, execution authority and feedback under particular operating conditions.

3. Core Characteristics and Capabilities of Agentic AI

Section 2 established Agentic AI as a systems paradigm rather than a single model or software component. The next question is functional: what must such a system be able to do? The literature repeatedly identifies goals, perception, reasoning, planning, memory, action, reflection, adaptation, and coordination as central to sustained agency.

3.1 Autonomy and Goal-Directedness

Autonomy describes the extent to which a system can select and execute actions without continuous human specification. In Agentic AI, autonomy is closely linked to goal-directedness because independent action must remain organized around an intended outcome. Goal-directed systems maintain objectives beyond a single input-response cycle. Work on behavioral flexibility in agent architectures similarly emphasizes adapting behavior while maintaining goal orientation

(Saadi et al., 2020). Long-horizon agents may need to preserve a high-level intention while changing intermediate actions as new information becomes available (Bidochko & Vyklyuk, 2026).

3.2 Goal Management and Task Decomposition

Complex objectives usually cannot be executed as single actions. Agentic systems therefore require mechanisms for decomposing goals into manageable subtasks, maintaining dependencies, tracking completion, and revising priorities. MetaGPT demonstrates this principle through role-based decomposition of complex tasks into coordinated responsibilities (Hong et al., 2024). The distinction between goal management and planning is important: goal management determines what remains to be achieved, while planning determines which actions should achieve it.

3.3 Perception and Environmental Interaction

Agentic behavior requires information about the environment in which action occurs. The relevant inputs are increasingly multimodal and may include text, structured records, images, audio, API responses, software states, or physical sensor data. PaLM-E integrates language with perceptual inputs for environment-grounded decision processes (Driess et al., 2023). RT-2 similarly connects visual-language representations with robotic action (Brohan et al., 2023).

3.4 Context and World-State Representation

Raw perception is insufficient for sustained agency. An agent also requires a representation of its current situation, including goals, recent actions, available resources, constraints, unresolved tasks, and relevant environmental conditions. Language-conditioned robotic systems such as SayCan combine semantic reasoning with information about available physical actions (Ahn et al., 2022). Context and world-state representation therefore provide the bridge between perception, memory, reasoning, and planning.

3.5 Reasoning and Decision-Making

Reasoning transforms goals, state, knowledge, and observations into judgments about what should happen next. Chain-of-Thought prompting improves multi-step reasoning by eliciting intermediate steps (Wei et al., 2022). Tree of Thoughts expands this process by exploring multiple candidate reasoning states before selecting a trajectory (Yao et al., 2023b). Language

Agent Tree Search combines language-model reasoning with evaluation and tree search (Zhou et al., 2024).

3.6 Planning and Replanning

Planning organizes future actions toward an objective. Replanning modifies those actions when the environment, available information, or previous assumptions change. LLM+P separates semantic interpretation from formal planning by using a language model to construct a planning representation that can be solved by a classical planner (Liu et al., 2023a). Planning in Agentic AI is therefore a recurrent process: goal, plan, action, observation, evaluation, and continuation or replanning.

3.7 Memory and Knowledge

Memory provides continuity across agent interactions. It allows information acquired at one stage to influence later reasoning, planning, and action. Architecturally, the most useful distinction is functional: working/contextual, episodic, semantic, procedural, and shared/distributed memory.

3.7.1 Working/Contextual Memory

Working memory maintains information required for the current task, including active goals, recent observations, intermediate outputs, and short action histories. MemGPT treats active context as a limited resource and manages information between immediate and external storage (Packer et al., 2023).

3.7.2 Episodic Memory

Episodic memory stores records of previous interactions, actions, outcomes, and experiences. Its usefulness depends on retrieval quality. Recent cognitive memory architectures incorporate explicit retrieval and consolidation mechanisms (Westhaußer et al., 2025).

3.7.3 Semantic Memory

Semantic memory stores factual and conceptual knowledge that is not tied to one specific interaction episode. Self-RAG integrates retrieval with generation and self-evaluation so that evidence acquisition can become conditional on the system’s reasoning state (Asai et al., 2024).

3.7.4 Procedural Memory

Procedural memory stores reusable information about how to perform tasks. Voyager demonstrates this principle through a skill library that retains useful behaviors acquired during interaction (Wang et al., 2023).

3.7.5 Shared/Distributed Memory

Multi-agent systems extend memory beyond the individual agent. G-Memory develops hierarchical memory specifically for multi-agent systems (Zhang et al., 2025). Persistent context-sharing protocols similarly address how state can move between autonomous agents while preserving continuity (Masoor, 2025).

3.8 Action and Tool Use

Action converts internal decisions into effects on the external environment. Tool use expands the effective capability of the reasoning model. Gorilla investigates large-scale API interaction and tool selection, while ToolLLM focuses on enabling models to interact with extensive real-world API collections (Patil et al., 2023; Qin et al., 2023). Tool use therefore forms the execution boundary between cognition and environment.

3.9 Reflection, Evaluation, and Self-Correction

Reflection allows an agent to evaluate its intermediate or completed behavior and use that evaluation to influence subsequent decisions. Reflexion stores linguistic feedback from previous attempts and reuses it during later trials (Shinn et al., 2023). Self-Refine similarly organizes iterative generation around feedback and revision (Madaan et al., 2023).

3.10 Learning and Adaptation

Learning and adaptation allow agent behavior to change in response to experience or environmental conditions. Research on lifelong LLM-based agents increasingly treats perception, memory, and action as connected components of continual adaptation (Zheng et al., 2026). Agentic adaptation therefore ranges from immediate contextual adjustment to persistent cross-task learning.

3.11 Communication, Coordination, and Orchestration

Communication enables agents or components to exchange information. Coordination aligns their activities, while orchestration determines how tasks, roles, and execution sequences are organized. CAMEL demonstrates role-based communication among language-model agents (Li et al., 2023). MetaGPT organizes specialized roles around structured collaborative workflows (Hong et al., 2024). AgentVerse investigates dynamic multi-agent collaboration and emergent interaction (Chen et al., 2023).

3.12 Long-Horizon Persistence

Long-horizon persistence refers to the ability to maintain coherent goal pursuit across extended sequences of actions. Thought-management architectures for long-horizon LLM agents address this need by organizing evolving reasoning state and goals across extended interaction sequences (Bidochko & Vyklyuk, 2026).

3.13 Capability Integration and Emergent Agentic Properties

Autonomy should not be represented as a component equivalent to memory, planning, or tool use. There is no single architectural module that independently produces autonomy. An agent becomes operationally autonomous when goals, state, reasoning, planning, memory, action, feedback, and adaptation are connected into a functioning loop. Planning alone does not create autonomy because a plan still requires an objective, current state, execution mechanism, and feedback. Tool access alone does not create autonomy because the system must determine when and why a tool should be used. Autonomy is therefore better interpreted as a higher-order property generated by sustained coupling among agentic capabilities.

Table 5. Functional capability taxonomy of Agentic Artificial Intelligence.

Functional group	Principal responsibility	Representative mechanisms	Key dependency
1. Goal and context management	Maintain objectives and constraints	Intentions; goal predicates; task records	State and completion criteria
2. Perception and state representation	Acquire and represent observations	APIs; sensors; belief states; world models	Grounding and information quality

Functional group	Principal responsibility	Representative mechanisms	Key dependency
3. Reasoning and decision-making	Evaluate alternatives and select decisions	Logic; search; learned policies; LLM inference	Knowledge and constraints
4. Planning and replanning	Organize and revise actions	PDDL; decomposition; generated plans	Goals, action models and feedback
5. Memory and knowledge	Retain, retrieve and update information	Belief bases; retrieval; episode and skill stores	Consistency and provenance
6. Action and tool use	Execute operations	API calls; software tools; actuators	Permissions and execution status
7. Evaluation and reflection	Assess outputs and behavior	Tests; critics; verifiers; rewards	Independent evidence and criteria
8. Learning and adaptation	Update behavior or knowledge	Replanning; feedback memory; policy learning	Valid feedback and version control
9. Coordination and orchestration	Organize execution responsibilities	Schedulers; workflows; supervisors	Shared state and communication

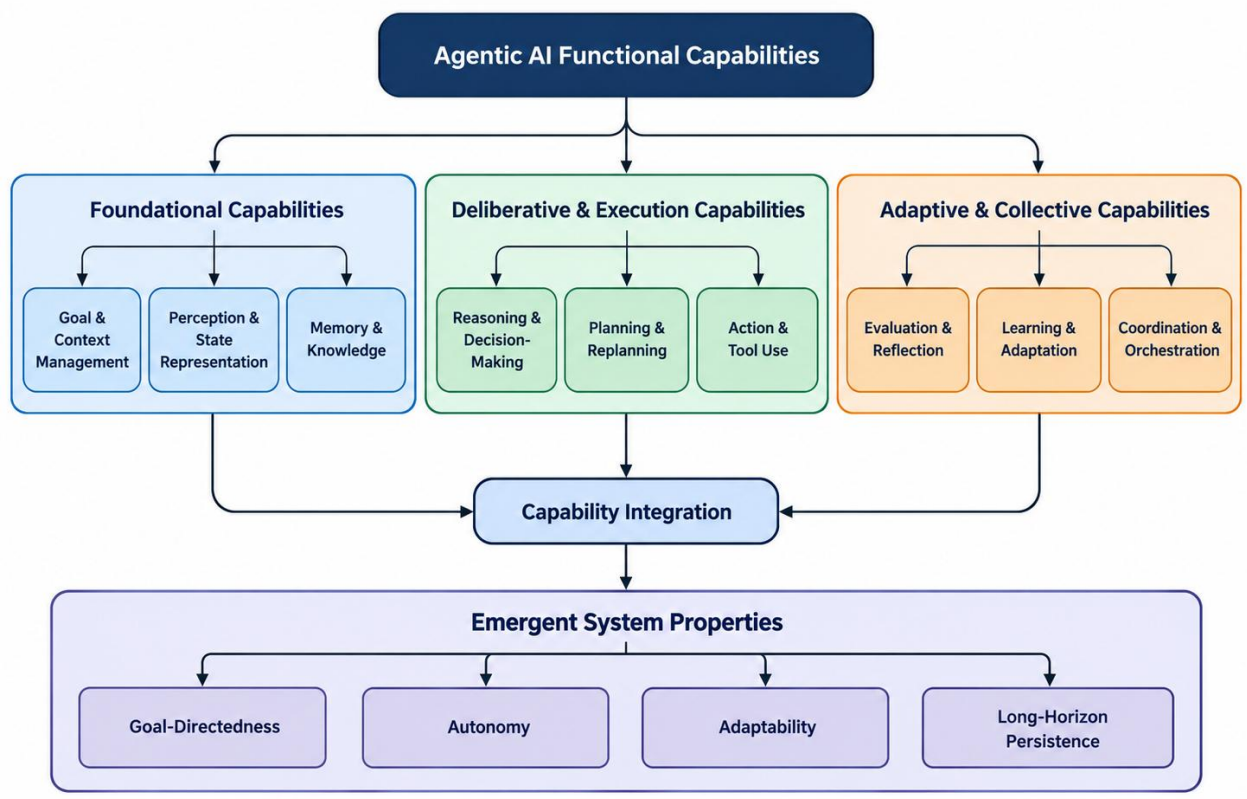


Fig. 2. Hierarchical taxonomy of Agentic AI functional capabilities and emergent system properties. Conceptual visualization prepared with generative-AI assistance; design and scientific content require author verification.

The nine capability groups are analytical families rather than required standalone modules. One model can implement more than one function, and one function may be distributed across several specialized components. Explicit coordination among multiple agents is not required by every implementation, although coordination among internal components can still be relevant to single-agent systems.

Autonomy is an emergent operating property constrained by authorization and intervention. Goal-directedness requires coherence between the system's active objective and its decisions. Adaptability concerns appropriate behavioral changes, not merely the presence of a learning component. Long-horizon persistence requires consistent goal and state continuity across extended actions. The revised taxonomy shown in Fig. 2 separates these properties from the nine function groups and avoids inferring demonstrated effectiveness from component names.

4. Architectural Paradigms and Design Approaches for Agentic AI

Section 3 identified the capabilities associated with Agentic AI. The next analytical step is architectural: how are these capabilities implemented, connected, and controlled? Similar capabilities can be realized through different mechanisms. This section addresses the first part of RQ1.3 by examining symbolic/classical, neural/generative, hybrid neuro-symbolic, and multi-agent architectures.

4.1 Capability versus Architecture: Establishing the Analytical Distinction

A central distinction in this survey is that capabilities describe what an Agentic AI system can do, whereas architectural mechanisms describe how those capabilities are implemented and coordinated. Reasoning is a capability. Logical inference, language-model inference, search, and distributed deliberation are realization mechanisms. Memory is a capability, while context windows, vector databases, knowledge bases, and shared stores are architectural mechanisms. Planning is a capability, while PDDL, tree search, planner-executor structures, and multi-agent delegation are possible implementations.

Four complementary dimensions provide a more useful description than one mutually exclusive classification. Computational realization identifies how a function is calculated; organizational topology identifies who or what holds responsibility; control structure determines how execution transitions occur; and interface organization specifies where state and environmental actions cross component boundaries. An architecture can be simultaneously neural in reasoning, symbolic in action validation and hierarchical in delegation.

A capability label should be assigned on the basis of described behavior, not marketing terminology. For example, a component labelled a planner might construct an explicit constrained plan, generate informal steps, or merely select the next tool call. These alternatives represent different mechanisms despite their shared functional label.

4.2 Symbolic and Classical Agent Architectures

Symbolic and classical architectures represent knowledge, goals, actions, and state through explicit structures. Their reasoning processes commonly operate over rules, logical expressions, plans, beliefs, or symbolic state transitions.

4.2.1 Rule-Based Architectures

Rule-based architectures encode behavior through explicit condition-action or inference rules. Early expert-system research demonstrated the usefulness of explicit rule structures for representing domain knowledge (Swartout, 1985). In agentic settings, similar mechanisms can support deterministic routing, policy enforcement, task constraints, and safety conditions.

4.2.2 BDI Architectures

Belief-Desire-Intention architectures separate internal state into beliefs, desires, and intentions. This separation provides a structured control cycle for maintaining goal commitment while updating beliefs as conditions change (de Silva et al., 2020).

4.2.3 Classical Planning Architectures

Classical planning architectures model an initial state, goal conditions, available actions, and the effects of those actions. A planner searches for a valid sequence that moves the system toward the desired state. PDDL-based research investigates how pretrained language models can interact with explicit planning representations (Silver et al., 2022).

4.2.4 Cognitive Architectures

Cognitive architectures integrate memory, decision processes, goals, learning, and action within a persistent computational structure. Recent work on cognitive architectures for language agents revisits this systems perspective in the context of foundation models (Sumers et al., 2023).

4.3 Neural and Generative Agent Architectures

Neural and generative architectures place learned models, particularly foundation models, at the center of reasoning and control. Surveys of LLM-based intelligent agents increasingly describe systems composed of foundation models, memory, tools, planning, and environmental interfaces (Cheng et al., 2024).

4.3.1 Foundation-Model-Centered Agents

A foundation-model-centered architecture uses an LLM or multimodal foundation model as the principal reasoning component. Supporting components provide memory, tools, retrieval, execution, and state management. The model itself is therefore not equivalent to the complete agentic system.

4.3.2 ReAct Architectures

ReAct combines reasoning and acting within a repeated interaction loop. The model generates an intermediate reasoning step, performs an action, receives an observation, and continues until completion (Yao et al., 2023a).

4.3.3 Planner-Executor Architectures

Planner-executor architectures separate task organization from action execution. A planner decomposes the objective and maintains the task structure, while an executor carries out selected actions. This separation improves modularity because planning can be revised without replacing the execution mechanism.

4.3.4 Reflection/Critic Architectures

Reflection architectures add an evaluative component to the reasoning-action loop. CRITIC combines model-generated critique with external tools for checking and revision (Gou et al., 2024).

4.3.5 Tool-Augmented Architectures

Tool-augmented architectures connect foundation models to external computational resources. ChemCrow demonstrates this pattern by integrating specialized chemistry tools with a language model (Bran et al., 2023).

4.3.6 Retrieval-Augmented and Agentic RAG Architectures

Conventional Retrieval-Augmented Generation uses retrieved external information to augment model context. Agentic RAG extends this pattern by making retrieval itself part of the decision process. Singh et al. (2025) characterize Agentic RAG as the integration of agentic workflows with retrieval to support more adaptive information acquisition.

4.4 Hybrid Neuro-Symbolic Architectures

Hybrid neuro-symbolic architectures combine learned neural mechanisms with explicit symbolic representations, rules, constraints, or planners. Recent work describes neuro-symbolic Agentic AI as a distinct integration problem involving architecture, interfaces, and responsibility allocation across these mechanisms (Hakim et al., 2026). Neuro-symbolic multi-agent reinforcement learning similarly combines probabilistic learning with explicit symbolic structure

(Subramanian et al., 2024). TB-CSPN provides another hybrid architecture in which semantic processing is combined with structured Petri-net-based control (Borghoff et al., 2025).

Neuro-symbolic integration is most informative when an explicit computational interface links learned interpretation to formal operators, rules or constraints. In LLM+P, a language-model component helps translate task descriptions into a planning representation and a separate planner computes the solution (Liu et al., 2023a). The major assurance challenge is translation validity: a formally correct solution to an incorrectly encoded problem remains unsuitable for the original objective.

Other hybrid patterns include neural perception coupled to a symbolic state model, model-generated candidate actions filtered by an external checker, and structured workflow controllers operating over language-model-produced information. None is universally superior; their benefits must be assessed in terms of representation alignment, integration cost and independently verified outcomes.

4.5 Multi-Agent Architectures and Organizational Structures

Multi-agent architectures distribute computation across interacting agents. This architectural category is different from the symbolic-neural distinction. Multi-agent describes organizational structure, whereas symbolic, neural, and hybrid describe computational realization. Recent surveys of LLM-based collaboration identify task allocation, communication, role specialization, shared context, and coordination as central design dimensions (Tran et al., 2025).

4.5.1 Hierarchical/Supervisory Architectures

Hierarchical architectures place a supervisor, manager, or orchestrator above specialized workers. The supervisory component maintains the global objective, decomposes work, delegates tasks, and integrates outputs.

4.5.2 Horizontal/Decentralized Architectures

Decentralized architectures distribute authority among peer agents. This organization can improve parallelism and reduce single points of failure, but increases the difficulty of maintaining consistent state and resolving conflict.

4.5.3 Sequential/Workflow Architectures

Sequential architectures arrange agents or components into an ordered workflow. Agentic workflow research shows how structured multi-step patterns can organize LLM systems beyond isolated prompting (Singh et al., 2024).

4.5.4 Role-Specialized Architectures

Role-specialized architectures assign agents distinct responsibilities according to expertise or task function. Chain-of-Agents coordinates multiple agent roles within a broader agent foundation-model approach (Li et al., 2025a).

4.5.5 Debate/Critic/Evaluator Structures

Debate and critic architectures allocate evaluation to multiple interacting agents. Multi-agent debate has been studied as a mechanism for improving factuality and reasoning through interactions among model instances (Du et al., 2024).

4.5.6 Hybrid Coordination

Practical multi-agent architectures often combine supervisory, decentralized, sequential, and role-specialized patterns. Recent work on systematic design of LLM-based multi-agent systems emphasizes that architecture must specify interaction structure rather than treating a collection of agents as sufficient by itself (Zhou et al., 2026).

4.6 Memory and Knowledge Architectures

Memory design constitutes an architectural subsystem because it determines where state is stored, how it is retrieved, and which components may modify it. Declarative approaches to task-oriented agents show one direction by making knowledge structures more explicit and controllable (Joshi et al., 2025). Persistent context-sharing approaches address a related problem across multiple agents (Masoor, 2025).

4.7 Tool and Environment Interfaces

Tool and environment interfaces define how internal decisions become external actions. They also determine which parts of the environment are visible and actionable. Architecturally, the tool interface is both a capability boundary and a control boundary.

4.8 Orchestration and Control Structures

Orchestration determines how components, tools, and agents are activated and sequenced. It defines who controls the next transition in the workflow. Derouiche et al. (2025) identify architectures, protocols, and design choices as central to contemporary agentic frameworks.

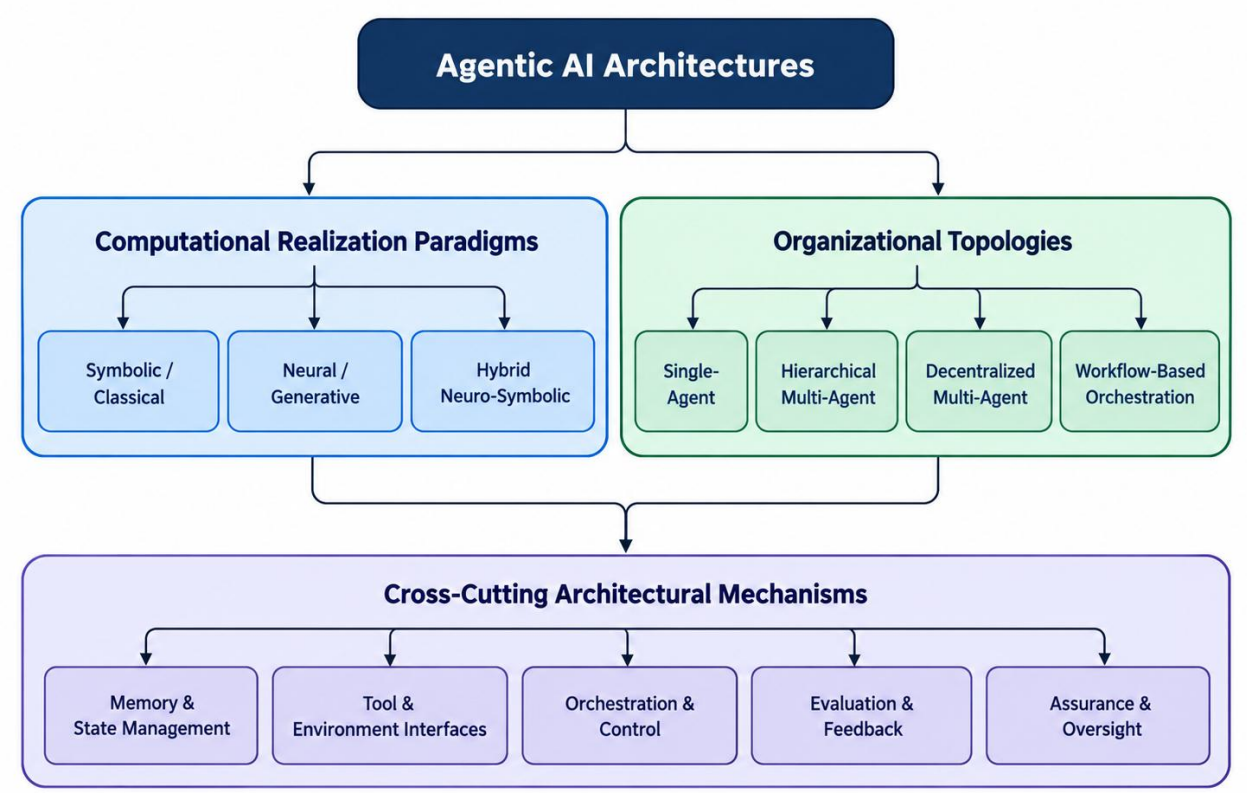


Fig. 3. Computational realization paradigms, organizational topologies, and cross-cutting architectural mechanisms of Agentic AI. Conceptual visualization prepared with generative-AI assistance; design and scientific content require author verification.

Orchestration denotes the control responsibility that routes tasks, invokes components, transfers authority, handles exceptions and terminates execution. It is not a fourth computational paradigm. A deterministic state machine can govern a neural decision component, and a symbolic planner can operate inside a multi-agent workflow. The architecture should make responsibility for retries, error reporting, state changes and human escalation explicit.

The classification summarized in Table 6 and Fig. 3 therefore describes independent but interoperating dimensions. Its purpose is comparison of actual control and realization choices, not assignment of a single technology label to a complex agentic system.

Table 6. Principal dimensions for classifying Agentic AI architectures.

Dimension	Representative alternatives	Guiding question
Computational realization	Symbolic, neural, hybrid neuro-symbolic	How is a capability computed?
Organizational topology	Single-agent, hierarchical, decentralized	How is responsibility distributed?
Control structure	Rules, state machines, planners, model-mediated	How are execution transitions determined?
State and memory	Local, external, shared, distributed	How are records retained and updated?
Environmental interface	APIs, software tools, physical actuators	How do decisions produce actions?
Assurance and oversight	Constraints, verification, authorization	How is authority bounded and traced?

5. Comparative Analysis, Representative Systems, and Evaluation of Agentic AI Architectures

Section 4 showed that similar agentic capabilities can be implemented through different computational structures. The purpose of this section is not to identify a universally preferable architecture, but to compare how architectural paradigms represent, coordinate, and sustain agency. The literature consequently points toward architecture as a systems-level design problem rather than a model-selection problem (Miehling et al., 2025; Wissuchek & Zschech, 2025).

5.1 Architectural Comparison Dimensions

Architectural comparison requires dimensions that remain meaningful across paradigms. Thirteen dimensions are used here: goal representation, reasoning, planning, memory, control flow, tool use, adaptability, reflection, coordination, interpretability, human oversight, scalability, and reliability.

Goal representation concerns how objectives, priorities, and completion conditions are maintained. Symbolic systems typically represent goals explicitly, whereas neural systems often

encode them through prompts, context, or structured state. Hybrid systems can translate natural-language goals into symbolic constraints. Multi-agent systems add a distinction between global objectives and agent-specific subgoals.

Reasoning and planning reveal a similar divergence. Symbolic systems use explicit rules, search, constraints, or state-transition models. Neural systems depend more strongly on foundation-model inference, generated decomposition, or search over model-generated candidates. Hybrid architectures combine these mechanisms. Multi-agent systems may distribute reasoning and planning across specialized agents.

Memory differs in representation and ownership. Symbolic architectures may maintain beliefs, explicit state, or knowledge bases. Neural agents often use active context, retrieval, vector stores, or external episodic memory. Hybrid architectures combine structured and learned memory. Multi-agent systems additionally require choices concerning private, shared, and distributed state.

Control flow concerns who or what determines the next action. It may be encoded through rules, planners, state machines, workflow graphs, model-generated decisions, or agent supervisors. Tool use similarly varies from predefined operators to dynamic API selection and role-specific tool access.

Adaptability and reflection distinguish architectures by how they respond to change and error. Symbolic systems commonly rely on state revision or replanning, while neural systems can revise behavior through context, critique, retrieved experience, or model-generated feedback. Hybrid architectures can pair adaptive neural behavior with formal checking. Multi-agent systems may externalize reflection through critic or verifier roles.

Interpretability and human oversight depend strongly on architectural structure. Explicit rules, goals, and state transitions are generally easier to inspect than stochastic model-generated decisions. Neural systems therefore depend more heavily on logs, provenance, monitoring, and external validation. Scalability and reliability complete the comparison. Larger architectures can improve specialization and parallelism, but also introduce more state, communication, coordination, and synchronization requirements.

Comparison must separate architectural specification from performance evidence. Goal representation, reasoning, planning, memory, control flow, tool interaction, adaptation,

evaluation, coordination, interpretability, oversight, scalability and reliability can be described using implementation information. Whether these mechanisms work consistently requires evaluation under appropriate conditions. A more transparent model-generated explanation is not automatically a more faithful one; an explicit symbolic rulebase is not automatically complete or correct.

For systems whose decisions alter external resources, action proposal, authorization, attempted execution and observed outcome should be reported separately. This distinction makes it possible to attribute a task failure to invalid planning, a tool-interface problem or an inaccurate account of the completed action.

5.2 Cross-Paradigm Capability Realization

Table 7. Cross-paradigm architecture-capability realization matrix.

Capability	Symbolic/classical	Neural/generative	Hybrid	Organization
Goals	Predicates, intentions	Contextual instructions	Neural interpretation + constraints	Central or delegated
Perception/state	Belief states	Learned multimodal representations	Perception + structured state	Local or distributed
Reasoning	Logic, rules, search	LLM inference, learned policy	Model + symbolic reasoning	Individual or collective
Planning	Formal search, HTN	Generated plans	LLM+P, constrained search	Central or delegated
Memory	Structured knowledge	Context and retrieval	Learned + symbolic stores	Private or shared
Action/tools	Operators, predefined tools	Model-selected tools	Generated action + checks	Local or delegated
Evaluation	Logical checks,	Critics and	Generated	Central or

Capability	Symbolic/classical	Neural/generative	Hybrid	Organization
	tests	revisions	critique + verification	independent
Learning	Belief/policy revision	Experience-conditioned behavior	Learned updates + constraints	Local or coordinated
Coordination	Protocols and schedulers	Model routing	Structured + learned control	Single, hierarchical or peer

The matrix reveals that no capability maps to only one architecture. Planning can be formal, generative, hybrid, or distributed. Memory can be explicit, retrieval-based, mixed, or shared. Reflection can be internal, externally verified, or delegated to another agent. Multi-agent systems change the distribution of capability rather than introducing a single new reasoning or planning principle.

Table 7 is a capability-realization matrix, not an empirical ranking. The same planning function may arise from symbolic search, a foundation-model-derived sequence of candidate steps or a hybrid semantic/formal pipeline. The coordination column describes where responsibility is held rather than an independent planning algorithm. Similar distinctions apply to memory stores, tool interfaces and evaluators. Each cell requires representative primary evidence when used for system-specific claims.

5.3 Architectural Strengths, Limitations, and Trade-Offs

The evidence does not support a universal architecture for all forms of Agentic AI. Symbolic architectures provide explicit representations of goals, state, actions, and constraints, supporting traceability and predictable control, but requiring substantial representational effort. Neural architectures are well suited to natural-language interpretation, unstructured information, semantic generalization, and dynamic tool interaction, but introduce stochastic behavior, weaker intrinsic traceability, prompt sensitivity, and variable planning reliability (Xu, 2026).

Hybrid architectures attempt to combine neural flexibility with symbolic structure or verification. The main challenge is representation alignment across heterogeneous components (Hakim et al., 2026). Multi-agent architectures provide specialization, modularity, parallel execution, and distributed critique, but each additional agent introduces communication, synchronization, and responsibility relationships. Recent systematic-design work similarly emphasizes that multi-agent quality depends on interaction architecture rather than merely the number of agents (Zhou et al., 2026).

Architectural selection should therefore be condition-dependent. Relevant factors include environmental uncertainty, task horizon, regulatory requirements, tool complexity, need for formal guarantees, interpretability, latency constraints, and coordination demands.

Architectural trade-offs should be assessed as conditional rather than absolute. Explicit symbolic operators support inspection only to the extent that their action model reflects the environment. Learned inference is flexible with unstructured language but can generate actions based on false assumptions. Hybrid integration creates useful division of responsibility but introduces translation errors. Additional agents may permit parallelism yet increase state conflicts and communication overhead. Context and task constraints determine which trade-offs are acceptable.

5.4 Representative Agentic AI Systems and Architectural Case Studies

BDI-style architectures represent a structured deliberative pattern in which beliefs, goals, intentions, and plans remain explicit. AgentLite illustrates a modular architecture for task-oriented LLM agents in which model reasoning is combined with supporting components for actions, tools, and workflow management (Liu et al., 2024). TB-CSPN represents a contrasting design in which model-based semantic processing operates within a more formally structured control architecture (Borghoff et al., 2025). Hierarchical multi-agent architectures use a central supervisor to maintain the global objective, assign subtasks, and aggregate outputs, while decentralized architectures distribute decision authority across peers. Graph-based systems encode workflow transitions explicitly, separating control topology from the reasoning model.

Representative systems illustrate distinct units of analysis. STRIPS specifies action representation and search, whereas BDI describes a more complete deliberative agent structure (Fikes & Nilsson, 1971; Rao & Georgeff, 1995). ReAct connects model-generated reasoning

with iterative tool interaction (Yao et al., 2023a); LLM+P distributes the planning function between semantic interpretation and formal planning (Liu et al., 2023a). Reflexion records linguistic feedback, and Voyager retains executable skills for subsequent tasks (Shinn et al., 2023; Wang et al., 2023).

MetaGPT and AutoGen demonstrate role-based and configurable conversational coordination, respectively (Hong et al., 2024; Wu et al., 2024). Neither use of multiple agent labels nor use of language-model dialogue establishes that individual contributors are independent, effective or reliable. Table 8 lists the principal mechanism supported by each source without attempting an unsupported cross-system performance ranking.

A worked planning comparison clarifies the paper's principal contribution: STRIPS computes an action plan over explicit state and operators; ReAct incrementally selects model-mediated actions informed by observations; LLM+P combines neural language interpretation with a formal planner; and MetaGPT distributes task-planning responsibilities across specialized roles. These are different realizations and organizations of planning, not interchangeable implementations. Evidence for their suitability requires matched evaluation conditions.

Table 8. Representative agentic systems and capability-realization relationships.

System	Principal mechanism	Illustrated capability	Architectural lesson
STRIPS	Action operators + search	Planning	Explicit state/action semantics
BDI	Beliefs, desires and intentions	Goal management	Structured control and commitments
ReAct	Interleaved model inference and actions	Reasoning, feedback, tools	Recurrent environmental loop
LLM+P	Language-to-PDDL + planner	Planning	Hybrid interpretation and computation
Reflexion	Linguistic feedback memory	Evaluation and adaptation	Behavioral correction without model retraining
Voyager	Reusable executable skill library	Procedural memory	Skill retention and retrieval
MetaGPT	Role-specialized	Multi-agent coordination	Distributed task

System	Principal mechanism	Illustrated capability	Architectural lesson
	workflow		responsibility
AutoGen	Conversational agent interactions	Communication and control	Configurable multi-agent organization

5.5 Evaluation Metrics and Benchmarking of Agentic AI Architectures

Agent performance is a property of an integrated configuration, not exclusively of the underlying language model. Task success may depend on model reasoning, planning, retrieved information, tool permissions, environmental state and execution policy. Benchmark comparisons must therefore identify the complete system under test. AgentBench studies interactive decisions, GAIA examines general assistant tasks, WebArena focuses on web-task execution, SWE-bench addresses software issue resolution, OSWorld assesses computer use, and τ-bench evaluates policy-constrained tool interactions (Liu et al., 2024; Mialon et al., 2024; Zhou et al., 2024; Jimenez et al., 2024; Xie et al., 2024; Yao et al., 2024).

MultiAgentBench provides a setting for examining collaborative and competitive multi-agent behavior, while AgentDojo addresses security under prompt-injection attacks against tool-using agents (Zhu et al., 2025; Debenedetti et al., 2024). None of these benchmarks on its own demonstrates comprehensive architecture quality. A high task-success score may coexist with opaque state changes, excessive resource use or weak resilience under changed conditions. Scores obtained with different tool sets, models, task versions or computational budgets should not be treated as a controlled architectural comparison.

Capability-oriented evaluation examines goal satisfaction, state accuracy, planning feasibility, memory retrieval and update consistency, action authorization, execution correctness, evaluator independence, effective adaptation and coordination overhead. Trajectory-oriented evaluation additionally examines failure accumulation, recovery, long-horizon continuity and unauthorized side effects. Architecture-oriented evaluation investigates how replacing a mechanism changes observed outcomes while holding other conditions as constant as reasonably possible. Repeated trials are important when stochastic inference contributes to decisions.

The present survey synthesizes representative evaluation methods without performing new controlled benchmark experiments. Its proposed taxonomy may support the design of future

ablations by identifying which computational responsibility or interface is varied, but its usefulness as an evaluation instrument has not yet been established.

Table 9. Representative benchmarks and their architectural evaluation scope.

Benchmark	Evaluation focus	Relevant capability	Scope limitation
AgentBench	Interactive decision-making	Reasoning/action	Does not isolate all component effects
GAIA	General assistant problems	Retrieval and tools	Answer quality is not execution assurance
WebArena	Web task execution	Planning, feedback	Web-based domain
SWE-bench	Repository issue resolution	Code editing and testing	Software engineering specialization
OSWorld	Desktop GUI interaction	Perception and execution	Computer-use domain
τ-bench	Policy-constrained tool dialogue	Tools and reliability	Simulated domain interactions
MultiAgentBench	Collaborative/competitive agents	Coordination	Selected multi-agent settings
AgentDojo	Prompt-injection resilience	Security and tool boundaries	Specified adversarial threats

5.6 Architecture Selection and Design Implications

Architecture selection should begin with functional and operational requirements rather than a favored model family or an assumption that more agents always provide greater capability. Formalized goals and stable constraints may justify explicit planners or state-machine control. Highly variable natural-language inputs may require learned interpretation, while tasks combining informal objectives with enforceable constraints may warrant neuro-symbolic integration. Systems operating over extended tasks may need explicit state checkpoints and persistent memory, and independently executable subtasks may benefit from delegation.

Selection also requires matching execution authority to operational consequences. A tool that only retrieves permitted information poses different risks from a component that updates

customer records or interacts with physical infrastructure. State ownership, trust boundaries, human approval, reversibility and failure recovery are therefore first-class architectural choices. Table 10 summarizes conditional approaches; it is not a universal recommendation hierarchy.

The strongest comparative studies would hold models, tools and task environments consistent while varying planning, memory or topology. Reporting additional latency, token use, error recovery and authorization failures together with successful task completion provides a more complete assessment than end-task accuracy alone.

Table 10. Task-dependent architectural design considerations.

Design requirement	Candidate approach	Primary consideration
Formal goals and constraints	Symbolic planning / structured control	Action-model correctness
Unstructured instructions	Foundation-model interpretation	Grounding and output verification
Natural-language formal planning	Neuro-symbolic integration	Correct translation to constraints
Extended continuity	Persistent memory and checkpoints	Provenance and recovery
Parallel subtasks	Role-based multi-agent organization	Coordination cost
Predictable workflow	State machine / workflow controller	Behavior under exceptions
Consequential actions	Authorization gate / human intervention	Execution rights and reversibility

6. Integrated Architecture-Capability Taxonomy and Unified Reference Architecture

Sections 3 to 5 established two recurring findings. First, Agentic AI architectures repeatedly exhibit similar functional capabilities, including goals, perception, reasoning, planning, memory, action, reflection, adaptation, and coordination. Second, these functions are realized differently across symbolic, neural, hybrid, and multi-agent systems. The central synthesis problem is therefore to preserve functional commonality without erasing architectural diversity. This section

addresses RQ1.4 by integrating these findings into an architecture-capability taxonomy and a paradigm-neutral reference architecture. Recent systems-oriented work similarly argues that Agentic AI requires analysis beyond individual models toward coordinated system structures (Miehling et al., 2025; Wissuchek & Zschech, 2025).

6.1 Synthesis Principles

The synthesis is guided by four principles. First, function and implementation should remain separate. Second, functional similarity does not imply architectural equivalence. Third, system properties should be distinguished from component capabilities. Fourth, the architecture must remain paradigm-neutral. This approach is consistent with emerging typological work that treats Agentic AI systems as configurations of architectural properties rather than as one uniform design (Wissuchek & Zschech, 2025).

The synthesis adopts six principles: distinguish function from implementation; permit many-to-many capability-mechanism mappings; separate computation from organization; distinguish functional responsibilities from emergent properties; preserve implementation neutrality; and make evidence traceable. A function should not be assigned merely because the system documentation uses familiar terms such as memory, reflection or autonomy. Wherever the implementation is not specified, uncertainty should be recorded.

6.2 Architecture-Invariant Agentic Capabilities

The comparison across paradigms supports nine recurring functional capability groups: goal and context management, perception and state representation, reasoning and decision-making, planning and replanning, memory and knowledge, action and tool use, evaluation and reflection, learning and adaptation, and coordination and orchestration. These capabilities are architecture-invariant only at the functional level. A single foundation model may perform several functions, while a multi-agent system may distribute one capability across several agents. Systems-engineering work similarly emphasizes that reliable AI behavior depends on specifications and interactions among components rather than model performance alone (Stoica et al., 2024).

The nine groups are stable only at the level of analytical description. Goal and context management, perception and state representation, reasoning and decisions, planning and replanning, memory and knowledge, action and tool use, evaluation and reflection, learning and adaptation, and coordination and orchestration do not correspond to nine mandatory deployed

services. Their realization may be combined, distributed or partially implicit within a particular architecture. The relevant analytical question is what operational responsibility is performed and by which mechanism.

6.3 Capability-to-Mechanism Mapping and Analytical Coding

The same capability can be realized through different computational mechanisms. For goal management, symbolic systems may use explicit goal predicates or intentions; neural systems commonly represent goals through prompts or contextual state; hybrid architectures can translate natural-language objectives into explicit constraints; and multi-agent architectures may distribute a global objective into agent-specific responsibilities.

For planning, the realization can be summarized as symbolic formal search or planning, neural-generated decomposition or action sequences, hybrid generative interpretation combined with symbolic planning or validation, and multi-agent delegated or distributed planning. Similar diversity appears in memory, action, reflection, and coordination. The many-to-many relationship between capabilities and mechanisms is central to the taxonomy.

For a reproducible conceptual mapping, the unit coded is a capability-realization instance, not an entire framework name. An instance specifies the function (for instance, plan generation), the responsible mechanism (such as explicit search or language-model inference), the state required, the action or output interface, and the source that describes the mechanism. A single system may receive multiple codes for one capability; one mechanism may serve multiple capabilities. Mechanism assignment should not be inferred solely from marketing terms such as “agent,” “memory,” or “autonomous.”

As an illustrative worked mapping, an explicit STRIPS planner implements plan synthesis over symbolic state and operators; a planner-executor LLM can generate and revise action sequences through language-conditioned inference; a hybrid system can translate user goals into symbolic constraints and verify a computed plan; and a multi-agent supervisor can delegate planning subtasks to individual components. These are distinct mechanisms that may coexist, not four mutually exclusive system types. Table 5 provides a cross-paradigm matrix; its cells describe representative patterns, not exhaustive or performance-ranked implementations.

A capability-realization instance is the unit of comparison. It records the functional capability, computational mechanism, organizational arrangement, state dependency, executable interface,

relevant constraints and supporting source. This coding structure is intentionally separate from a framework's name: two systems with the same model family can organize control differently, and two different models can rely on the same planning or retrieval mechanism.

The assignment rules are to identify functions from documented operations; describe mechanisms with the level of detail available; code topology independently; and distinguish directly documented relationships from author-inferred interpretations and insufficiently specified aspects. An unavailable description is not evidence that a function is absent. Table 11 sets out the fields and an illustrative planning record.

The mapping is many-to-many: $R \subseteq F \times M$, where F is the set of nine functions and M is the set of computational mechanisms. One mechanism can contribute to several functions and one function can require multiple mechanisms. This representation establishes an analytical vocabulary, not proof that the described system effectively performs every mapped responsibility.

Table 11. Analytical coding framework for capability-realization instances.

Coding field	Core question	Illustrative planning instance
Capability	What function is provided?	Planning
Mechanism	How is the function computed?	Symbolic state-space search
Organization	Where is responsibility allocated?	Single-agent controller
State dependency	Which information is required?	Goals, initial state and operators
Interface	What result or request is produced?	Plan of executable actions
Constraints	Which assumptions or controls matter?	Model validity, execution checks
Evidence status	How was assignment established?	Documented in primary source

6.4 Integrated Architecture-Capability Taxonomy

The evidence supports a multidimensional taxonomy. The first level contains realization paradigms: symbolic/classical, neural/generative and hybrid neuro-symbolic, with organizational topology represented separately composition. The second level contains architecture-invariant capabilities: goal/context management, perception/state, reasoning, planning, memory/knowledge, action/tool use, evaluation/reflection, adaptation, and

coordination/orchestration. The third level contains emergent properties: autonomy, goal-directedness, adaptability, and long-horizon persistence.

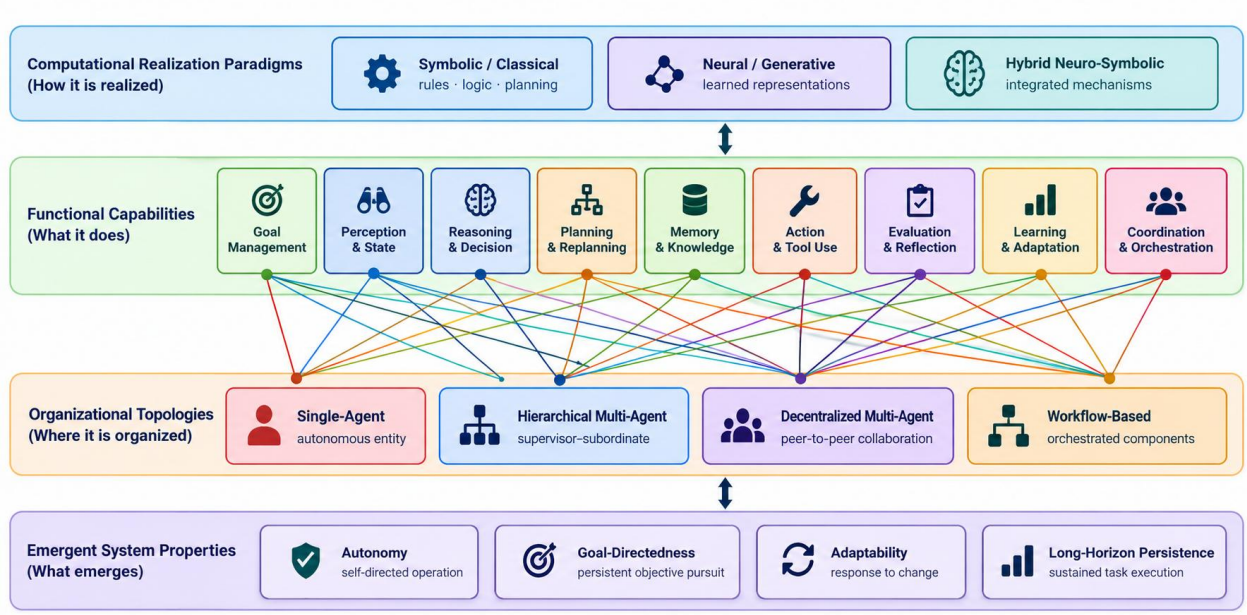


Fig. 4. Integrated architecture-capability taxonomy of Agentic Artificial Intelligence. Conceptual visualization prepared with generative-AI assistance; design and scientific content require author verification.

Four dimensions are retained in the integrated taxonomy: computational realizations, organizational topology, functional capabilities and emergent system properties. Their relationships are not a causal ladder: mechanisms contribute to functions; topology allocates responsibility; interfaces connect them; and the resulting behavior may exhibit autonomy, goal-directedness, adaptability and persistence. The crossed connections in Fig. 4 are conceptual, not quantitative evidence. Explicit symbolic checking and model-mediated inference can coexist inside both single- and multi-agent arrangements.

6.5 Formal Abstraction of an Agentic AI System

The illustrative interaction can therefore be understood as stateful recurrence: perceive current environment; reason and plan with goals and memory; act; observe the updated environment; evaluate the outcome; update memory; and, when enabled, adapt a policy or revise goals under orchestration and oversight. The symbols identify interfaces and conditional update roles rather than independent software modules or a mathematically proven control law. They do not

establish convergence, safety, reliability, or empirical superiority; these remain verification and evaluation problems.

The representation defines functional roles rather than mandatory software modules. A neural architecture may combine several functions within one model, a symbolic architecture may implement them through separate engines, and a multi-agent architecture may distribute them across agents.

$$A = \langle F, M, O, I, K, D \rangle \tag{1}$$

$$F = \{f_1, f_2, \dots, f_9\} \tag{2}$$

$$RFM \subseteq F \times M \tag{3}$$

$$o_t = P(e_t, u_t);\ (s_t, g_t) = G(s_{t-1}, g_{t-1}, o_t, m_t) \tag{4}$$

$$d_t = R(s_t, g_t, m_t;\ \theta_t) \tag{5}$$

$$\pi_t = \Pi(d_t, s_t, g_t, m_t;\ \theta_t) \tag{6}$$

$$\tilde{a}_t = A(C(\pi_t, s_t), s_t) \tag{7}$$

$$a_t = GateK(\tilde{a}_t, s_t, h_t) \tag{8}$$

$$(e_{t+1}, y_t) = T(e_t, a_t, \xi_t) \tag{9}$$

$$v_t = V(g_t, \pi_t, a_t, y_t, e_{t+1}) \tag{10}$$

$$m_{t+1} = \mu(m_t, o_t, a_t, y_t, v_t);\ \theta_{t+1} = L(\theta_t, m_{t+1}, v_t) \tag{11}$$

Here F denotes the nine functional groups, M the realization mechanisms, O the organizational topology, I the information and execution interfaces, K applicable constraints, and D the documented evidence. The relation RFM captures potentially many-to-many capability-mechanism associations. At step t, e is environment state, s operational state, g active goals, m memory, u external input, o observation, θ the model or policy state, d decision information, π a plan or action representation, h authority context, a executed action, y the observed result, and v the evaluation signal. P, G, R, Π, A, C, GateK, T, V, μ and L represent functional operators and not mandatory standalone software modules. The equations do not establish optimality, safety, convergence, or empirical validation.

6.6 Proposed Unified Reference Architecture

The proposed reference architecture contains a closed operational loop linking Environment/User/External Systems, Perception and State Representation, Goal and Context Management, Reasoning and Planning, Action and Tool Execution, and subsequent environmental outcomes. A Memory and Knowledge Plane connects perception, goals, reasoning, planning, and reflection. A Feedback, Reflection, and Adaptation loop evaluates outcomes and returns information to earlier stages. A Coordination and Orchestration Plane determines sequencing, routing, delegation, retries, escalation, and termination. Responsible, explainable, and safety assurance spans the complete system.

The proposed reference architecture is a conceptual integration device. Its planes should be treated as separable responsibilities, not necessarily separate deployed services. System builders should trace each responsibility to an implementable component, explicit state transitions, tool permissions, and measurable invariants. Comparative experiments and case studies are needed to determine whether this organization improves reliability, interoperability, or efficiency over alternatives.

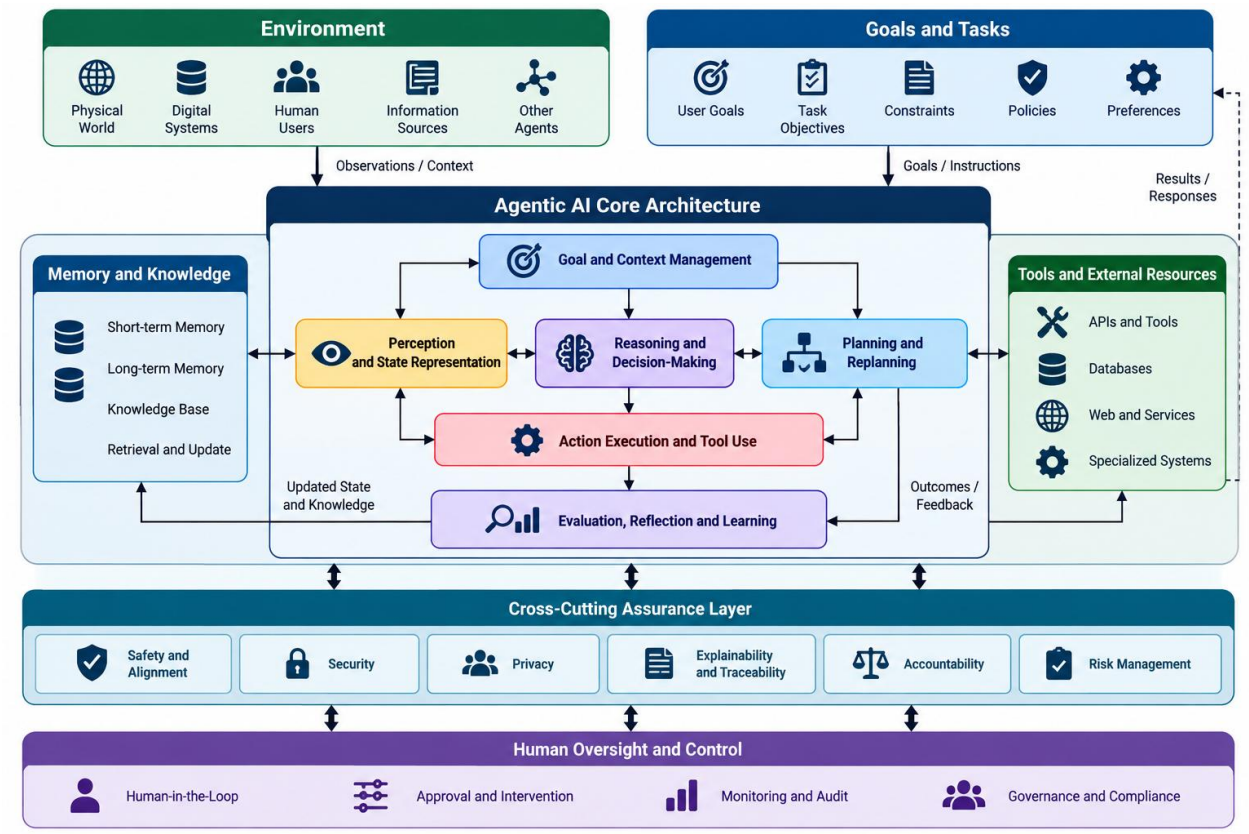


Fig. 5. Proposed unified reference architecture for Agentic AI: functional responsibilities, execution control, feedback, and cross-cutting assurance. Conceptual visualization prepared with generative-AI assistance; design and scientific content require author verification.

The unified reference architecture in Fig. 5 is a logical decomposition, not a mandated software stack. External observations enter a perception/state boundary; authorized objectives are maintained separately from untrusted source content; reasoning and planning produce action proposals; a trusted execution gate checks the proposed operation against relevant permissions; and the environment returns observable results. Memory supports relevant processes while evaluation informs state correction, replanning and controlled adaptation.

Coordination and orchestration define execution transitions for both internal components and multi-agent delegations. Assurance operates across planning, memory, tool execution and monitoring rather than exclusively at the end of the loop. The presence of an approval gate or evaluator cannot itself establish safety: specific controls require threat models, tests and operational evidence.

6.7 Illustrative Application and Assessment of the Proposed Framework

The coding framework can be applied to representative systems as an initial demonstration of its interpretive use. For STRIPS, the planning capability is realized through symbolic operators and state-space search; for BDI architectures, goal-related deliberation uses beliefs, desires and intentions; and for ReAct, reasoning and tool interaction are linked through recurrent language-model inference and environmental observations (Fikes & Nilsson, 1971; Rao & Georgeff, 1995; Yao et al., 2023a). These descriptions represent different scopes of architectural component and should not be collapsed into a single framework category.

For LLM+P, planning is distributed between learned language interpretation and an external formal planner. For Reflexion, retained natural-language feedback supports later evaluation and action revision, while Voyager emphasizes retention and reuse of executable skills (Liu et al., 2023a; Shinn et al., 2023; Wang et al., 2023). MetaGPT and AutoGen show how agent responsibilities may be organized across roles or conversational interactions without requiring a unique underlying computational paradigm (Hong et al., 2024; Wu et al., 2024).

A boundary case is a standalone model that outputs a proposed plan but cannot sustain stateful environmental action: it exhibits a planning-related output without necessarily constituting a complete agentic system. Another is a fixed multi-step pipeline that invokes tools without meaningful context-dependent decisions. These distinctions test the conceptual scope of the taxonomy but are not empirical validation.

Independent coders could evaluate classification clarity and agreement, and controlled studies could assess whether the mapping distinguishes meaningful changes in planning, memory or coordination. The proposed framework is therefore a testable synthesis instrument, not a verified reference standard.

Table 12. Illustrative capability-realization mapping of representative architectures.

Example	Functional responsibility	Realization mechanism	Organizational context
STRIPS	Planning	Symbolic operators and search	Standalone planner or agent
BDI	Deliberation and goals	Beliefs, desires, intentions	Individual agent

Example	Functional responsibility	Realization mechanism	Organizational context
ReAct	Reasoning/action	Model inference + tool interaction	Agent-environment loop
LLM+P	Planning	Neural interpretation + formal planner	Integrated pipeline
Reflexion	Evaluation/adaptation	Stored linguistic feedback	Agent with episodic memory
Voyager	Memory/adaptation	Executable skill library	Skill-enabled agent
MetaGPT	Coordination	Role-based workflow	Multi-agent
AutoGen	Communication/control	Conversational agent routing	Multi-agent

7. Explainability, Governance, and Responsible Architectural Design

The capabilities identified in previous sections increase the capacity of Agentic AI systems to pursue goals, maintain state, invoke tools, coordinate actions, and operate with reduced human intervention. These properties also increase the need for mechanisms that make autonomous behavior observable, constrainable, attributable, and secure. Responsible design is therefore treated as a cross-cutting architectural concern rather than as an isolated capability.

7.1 Explainability and Traceability

Explainability in Agentic AI extends beyond explaining a single prediction. An agent may interpret a goal, retrieve information, construct a plan, invoke tools, revise its strategy, communicate with other agents, and modify an external environment. Architectural traceability should preserve relevant information about goals, plans, retrieved evidence, tool invocations, environmental observations, inter-agent exchanges, and resulting actions. AI Chains illustrates how decomposition into inspectable intermediate steps can improve user visibility and control (Wu et al., 2022).

Explainability, traceability and auditability are different architectural objectives. Generated rationales may communicate a possible justification but are not a faithful record of every internal computation. Process-level evidence includes goal changes, retrieved sources, decision

proposals, verified tool invocations and observed outcomes. System-level explanations require locating responsibility across models, planners, memory stores and delegates. Preserving an action trace does not alone guarantee that a user can understand or contest the result.

7.2 Human Oversight and Controllability

Increasing autonomy does not eliminate the need for human control. Oversight can be embedded at goal specification, planning, action approval, tool execution, exception handling, or post-action review. Meaningful human control requires more than the nominal possibility of intervention (Santoni de Sio & van den Hoven, 2018). The architectural objective is bounded autonomy, in which agents operate independently within defined permissions while consequential or uncertain actions can trigger escalation.

Human oversight should specify the stages at which users or authorized operators may set goals, validate plans, approve actions, interrupt execution and review outcomes. Bounded autonomy need not be binary. Low-impact or reversible operations can sometimes proceed automatically under policy, whereas high-consequence actions may require explicit authorization. A human-in-the-loop label is inadequate if an operator receives no useful context or cannot intervene before an irreversible step.

7.3 Safety and Alignment Mechanisms

Safety mechanisms should constrain both what an agent intends to do and what it is technically permitted to execute. Formal-LLM integrates formal and natural language to constrain LLM-based agents (Li et al., 2024). TrustAgent incorporates explicit safety knowledge into planning and post-planning inspection (Hua et al., 2024). NeMo Guardrails provides programmable controls around model interaction (Rebedea et al., 2023).

Safety mechanisms must constrain executable authority rather than only text generation. Goal validation and constrained planning can screen proposed operations; tool gateways can enforce permission limits; independent tests can check selected properties; and monitors can identify deviations during execution. These controls are incomplete when the modeled environment omits relevant conditions or validation procedures fail. Assurance effectiveness must be evaluated rather than inferred from the presence of an extra critic or guardrail.

7.4 Accountability and Auditability

Accountability requires the ability to associate consequential actions with the system components, agents, policies, tools, and human decisions involved in producing them. Auditability provides the architectural evidence needed to support such attribution. Execution logs, tool-call histories, goal changes, memory operations, agent handoffs, policy interventions, and human approvals are particularly relevant. Explainability and accountability are therefore connected through the availability of evidence that permits responsible parties to understand and challenge system behavior (Doshi-Velez & Kortz, 2017).

Accountability combines organizational responsibility with technical evidence. A reliable audit record should distinguish who authorized an objective, which component proposed an action, which control approved it and whether the action occurred. NIST AI RMF, ISO/IEC 42001 and applicable AI regulation provide different kinds of organizational, management and legal requirements. Their presence in a bibliography does not demonstrate that a particular agent is compliant.

7.5 Privacy and Security

Agentic architectures expand the security boundary of conventional AI systems. An agent may maintain persistent memory, retrieve external information, access credentials, invoke tools, modify resources, communicate with other agents, and interact with users over extended periods. Permissions should therefore be attached to executable operations rather than inferred solely from natural-language outputs. Security should follow the architecture, with every boundary involving external input, memory access, tool execution, agent communication, or state modification treated as a trust boundary.

Persistent memory, tool invocation and inter-agent communication enlarge the attack surface. External pages, retrieved documents and tool responses must remain data sources, not automatically authoritative instructions. Indirect prompt injection exploits precisely this confusion (Greshake et al., 2023; Debenedetti et al., 2024). Principle-of-least-privilege access, protected credentials, validated tool arguments and independent execution authorization reduce some risks, but do not imply complete attack resistance. Sensitive information requires provenance, retention rules and access governance within both private and shared memory.

Table 13. Representative privacy and security threat boundaries in Agentic AI.

Boundary	Representative threat	Control consideration
External information	Indirect prompt injection	Instruction/data trust separation
Goals/planning	Unauthorized goal modification	Validation and policy constraints
Memory	Data exposure or poisoned records	Provenance and write authorization
Tool execution	Unpermitted operations	Least privilege and approval gates
Inter-agent interaction	Manipulated messages or delegation	Authenticated transfers and authority limits
Feedback/adaptation	Misleading evaluator information	Evidence validation
Audit and logging	Missing/altered trace evidence	Integrity-protected records

7.6 Responsible Design as a Cross-Cutting Architectural Assurance Layer

Explainability, oversight, safety, accountability, privacy, and security should not be implemented as isolated additions around an otherwise autonomous system. They interact with different components and stages of the agentic loop. The unified reference architecture is therefore extended with a cross-cutting assurance plane spanning goals, reasoning and planning, memory, action and tools, feedback and adaptation, and coordination.

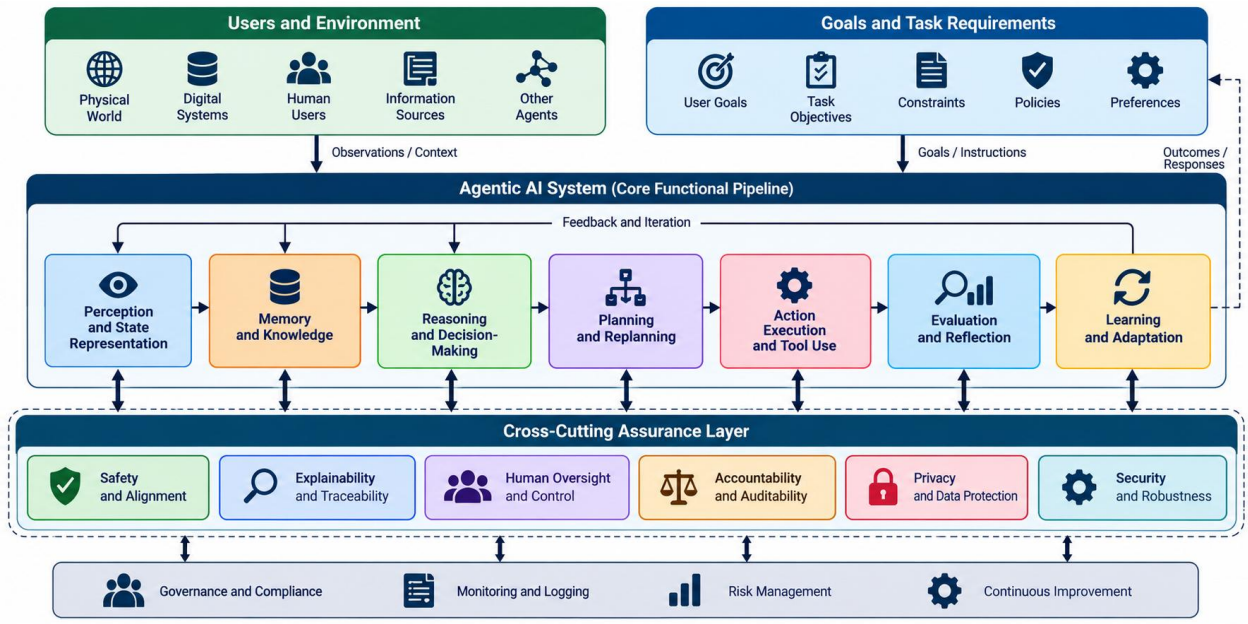


Fig. 6. Cross-cutting assurance layer for explainable, controllable, accountable, privacy-aware, and secure Agentic AI. Conceptual visualization prepared with generative-AI assistance; design and scientific content require author verification.

Table 14 and Fig. 6 locate assurance obligations at distinct trust boundaries. Goals require authorization; perceptions and retrieval require provenance; plans require constraint checks; memory requires controlled updates; execution requires enforceable permissions; evaluation requires credible evidence; and delegation requires traceable authority. The controls operate throughout a recurrent task, not as a one-time final review. Their implementation and measured effectiveness are separate research questions.

Table 14. Cross-cutting assurance responsibilities and observable evidence.

Agentic function	Assurance responsibility	Representative control	Observable evidence
Goals	Authorized objectives	Goal validation	Goal and approval records
Perception/state	Trusted information	Source and state checks	Provenance, state changes
Planning	Constraint-aware action	Formal checks, independent tests	Plan validation records
Memory	Integrity and permitted use	Access and update policies	Memory audit trail
Action/tools	Permissioned execution	Authorization gate	Tool calls and results
Evaluation	Reliable corrective feedback	Tests and independent verifiers	Assessments and revisions
Coordination	Accountable delegation	Role permissions, handoffs	Agent messages and handoff records
System governance	Ongoing risk management	Monitoring and human intervention	Incident and review evidence

8. Technical Challenges, Architectural Limitations, and Failure Modes

Agentic AI introduces failure modes that extend beyond model accuracy. Persistent goals, long reasoning chains, memory, tool execution, feedback, and multi-agent interaction create dependencies through which local errors can propagate across an entire task trajectory. These limitations should therefore be examined at the architectural level.

8.1 Goal Stability and Goal Alignment

Agentic systems require goals that remain coherent across planning, execution, feedback, and adaptation. Goal drift can occur when locally reasonable actions gradually diverge from the original objective. Model-level alignment can reduce harmful tendencies but does not guarantee trajectory-level alignment. Constitutional AI trains models to follow explicit principles, but agentic systems still require mechanisms that preserve those principles across multi-step action (Bai et al., 2022).

Goal drift can occur even when individual steps appear locally relevant. An agent pursuing a proxy such as maximizing test pass rates may violate the original purpose of preserving software correctness. Objective records, constraint monitoring and explicit approval of goal changes are possible mitigations, but depend on the adequacy of the initial goal specification.

8.2 Long-Horizon Reasoning and Planning

Long-horizon tasks require agents to preserve objectives, dependencies, state, and progress across extended action sequences. NaturalPlan demonstrates the continuing difficulty of natural-language planning for LLMs (Zheng et al., 2024). AgentBench similarly evaluates agents in interactive environments and exposes limitations in sustained reasoning, decision-making, and instruction following (Liu et al., 2023b).

Long-horizon error arises from interactions among inaccurate state, plan assumptions and observed execution outcomes. NaturalPlan and AgentQuest illustrate complementary perspectives on planning difficulty and trajectory monitoring (Zheng et al., 2024; Gioacchini et al., 2024). Checkpoints and replanning can improve recovery opportunities, but repeated plan generation based on the same false state may amplify rather than repair failures.

8.3 Memory Consistency and Context Degradation

Persistent memory improves continuity but creates risks involving stale state, retrieval error, conflicting information, and uncontrolled accumulation. The unresolved problem is not memory capacity alone, but maintaining a reliable evolving state whose provenance, relevance, and freshness can be established.

Persistent memory requires a managed lifecycle of collection, retrieval, updating, contradiction resolution and forgetting. Retrieval relevance is not equivalent to factual correctness or temporal validity. LongMemEval and MemBench highlight the need to test temporal updating, multi-session reasoning and memory efficiency. Distributed systems must additionally account for ownership of shared records and concurrent modifications.

8.4 Reflection and Verification Reliability

Reflection can improve agent behavior only when the evaluator correctly identifies errors. An agent that evaluates its own output may reproduce the same mistaken assumptions that generated the original result. Broader work on hallucination shows that model-generated information can remain unreliable even when systems include additional reasoning stages (Huang et al., 2025a). External tools, independent critics, formal verifiers, and multiple evaluators can strengthen verification, but introduce additional computation and may themselves fail.

Reflection, verification and recovery are distinct responsibilities. A model-generated critique may reproduce the original model's blind spots. Externally observable tests can provide more independent evidence for specified properties, but remain limited by test coverage. Successful recovery requires detecting the deviation, diagnosing its cause, applying a corrective action and verifying its effect.

8.5 Tool Reliability and External-System Dependence

Tool use expands agent capability while creating dependencies on systems outside the reasoning model. Tool-learning research identifies selection, parameterization, sequencing, and reliable execution as distinct technical problems (Qin et al., 2025). StableToolBench demonstrates that evaluation becomes unstable when live APIs change or fail (Guo et al., 2024).

An API request may fail while the underlying operation succeeds, creating ambiguity for retries. StableToolBench shows the impact of changing API availability and responses on evaluations

(Guo et al., 2024). Tool architectures should record request identifiers, attempted operations and observed status, and use idempotent or compensating operations where feasible. Not every external action can be reversed.

8.6 Multi-Agent Coordination and Emergent Failures

Multi-agent systems create specialization and parallelism but also introduce new paths for failure. Hammond et al. (2025) identify coordination failures, conflict, collusion, and emergent collective behavior among the risks of advanced multi-agent systems. MAEBE similarly treats emergent behavior as a system-level property requiring explicit analysis (Erisken et al., 2025).

Multi-agent failures include conflicting delegated goals, duplicate work, stale shared-state observations, inappropriate handoffs and correlated critic errors. Increased parallelism may reduce wall-clock time but raise communication costs. Reliability must therefore be evaluated through collective task outcomes, message integrity, synchronization and the consequences of distributed decisions, not only the output quality of each participant.

8.7 Interoperability and Standardization

Agentic systems increasingly depend on heterogeneous models, tools, memory stores, frameworks, enterprise systems, and other agents. Emerging protocols attempt to standardize these interfaces. A2A-oriented architectures address agent-to-agent communication and secure capability exchange (Habler et al., 2025). The remaining challenge is semantic and operational interoperability, not merely syntactic connectivity.

Protocols such as MCP and A2A address connectivity and capability exchange but do not establish shared semantics for task status, authority or execution success. Interoperability requires interface schemas, versioned capabilities, failure semantics, permission boundaries and consistent interpretation of state across independently designed components.

8.8 Scalability, Latency, and Computational Cost

Agentic execution often requires several model calls for planning, retrieval, reflection, verification, and coordination. Costs increase further when multiple agents communicate repeatedly. One research direction is greater use of smaller specialized models (Belcak et al., 2025). Efficiency also has an environmental dimension. Large-scale LLM use carries substantial lifecycle energy and carbon costs (Jiang et al., 2024; Liu & Yin, 2024).

Agentic computation includes repeated inference, retrieval, execution, verification and inter-agent communication. A cheaper system that fails often is not necessarily more efficient in successful-task terms. Resource allocation should be evaluated alongside task quality, reliability and environmental factors such as workload, hardware and energy source. Multi-agent scaling should not be inferred from agent count alone.

8.9 Security and Expanded Attack Surfaces

Agentic AI expands the security boundary because autonomous systems can read external content, maintain memory, invoke tools, execute code, communicate with other agents, and modify external state. Prompt Infection demonstrates that injected instructions can propagate between interacting LLM agents (Lee & Tiwari, 2024). More aggressive agent-based attacks demonstrate the broader risk of combining autonomy with extensive system permissions (Lupinacci et al., 2025).

Security analysis must extend from the input channel to persistent state, executable tool boundaries and agent-to-agent messages. Attacker-controlled content should not acquire instructions-level priority. Passing a fixed prompt-injection benchmark is evidence only under its defined scenarios, not proof of universal security. Runtime policy enforcement and incident traceability remain necessary.

8.10 Cross-Cutting Failure Propagation, Recovery, and Architecture-Level Evaluation

Agentic failures frequently cross capability boundaries. An incorrect retrieved state can affect planning, produce an inappropriate tool request, and then be reinforced when a weak evaluator records an apparent success. The component where the final error becomes visible may not be the component that introduced it. Architectural analysis should therefore distinguish an initiating deficiency, the propagation interface and the observed system consequence.

A useful failure-handling sequence separates detection, diagnosis, corrective action and verification. A failed tool invocation may be caused by invalid arguments, absent permissions or unavailable service infrastructure; repeating the operation is appropriate only for certain causes. Retrying a state-changing operation whose confirmation was lost can produce unintended duplication. Recovery policies must respect action reversibility, available execution evidence and the authority of the agent performing the correction.

Controlled failure-injection studies could introduce stale memory records, incorrect plan preconditions, conflicting multi-agent messages or unavailable tools. Outcomes should include error-detection accuracy, severity of downstream consequences, diagnosis quality, successful recovery, additional computational cost and violations of applicable constraints. The present study does not perform such experiments. Table 15 summarizes candidate mitigations, not experimentally proven guarantees.

Table 15. Technical challenges, failure mechanisms, and mitigation considerations.

Challenge	Failure mechanism	System consequence	Candidate mitigation
Goal instability	Ambiguous or drifting objectives	Proxy optimization or violation	Goal validation and constraint tracking
Long-horizon planning	Accumulated invalid dependencies	Failed extended execution	Checkpoints and state-based replanning
Memory inconsistency	Stale/conflicting state	Decisions on incorrect evidence	Versioning and provenance
Weak evaluation	Correlated critic errors	Undetected failures	Independent tests and criteria
Tool dependence	API failures, unclear status	Wrong or duplicated actions	Idempotency and status checks
Coordination failure	Conflicting agents or state	Duplication and deadlocks	Task contracts and synchronization
Interoperability	Incompatible semantics	Broken delegation and state	Schemas and semantic contracts
Efficiency	Excessive model/tool calls	Cost and latency increases	Selective routing and caching
Security	Untrusted instructions/authority	Unauthorized actions	Trust separation and least privilege

9. Open Research Challenges and Future Directions

The challenges identified in Section 8 indicate that the next stage of Agentic AI research requires more than improvements in foundation-model capability. Progress also depends on conceptual standardization, formalization, persistent memory, reliable planning, interoperable interfaces, architecture-level evaluation, and scalable coordination.

9.1 Terminological and Conceptual Standardization

Future work should move from broad descriptive definitions toward operational characteristics that can be observed or measured. Kasirzadeh and Gabriel (2025) characterize agents across autonomy, efficacy, goal complexity, and generality. Cihon et al. (2025) similarly investigate scalable measurement of agent autonomy through system inspection. A useful standard should distinguish minimum agentic conditions from advanced capabilities.

A useful definition should separate minimum properties of agentic operation from advanced optional capabilities. Independent experts could code representative systems under competing definitions and measure agreement. This would help identify whether ambiguous terminology, missing implementation details or substantive conceptual differences explain classification disagreement.

9.2 Formal Models of Agency

Most contemporary Agentic AI architectures are described through diagrams, prompts, workflows, and software components. Formalization should focus on observable architectural state and transitions. Goals, environment state, memory, actions, permissions, feedback, and coordination relationships can be modeled even when the internal operation of a foundation model remains probabilistic. Formal-LLM provides one direction for combining natural-language reasoning with formal constraints (Li et al., 2024).

Formal research should model inspectable states, action permissions, transitions and termination conditions even where neural inference remains probabilistic. Future studies can test whether independently specified invariants are preserved at execution boundaries and under changing environments. Such guarantees remain conditional on model coverage and assumptions.

9.3 Standardized Architecture Descriptions

Agentic systems are currently reported using inconsistent component names and levels of detail. Future research should establish a common architecture-description schema identifying goals, agents, models, memory stores, tools, control flow, state ownership, communication links, feedback paths, permissions, and assurance mechanisms. Stoica et al. (2024) argue that explicit system specifications are important for turning LLM-based systems into a more rigorous engineering discipline.

A common architecture specification should report functions, mechanisms, agents, goals, memory stores, execution interfaces, control flow, dependencies and assurance boundaries. Coding trials can assess reporting completeness, inter-rater agreement and coverage of unconventional designs. The objective is interoperability of descriptions, not a single mandated software architecture.

9.4 Reliable Long-Horizon Planning

Long-horizon agency remains limited by compounding errors, goal drift, context saturation, and weak progress estimation. Future architectures require stronger mechanisms for hierarchical decomposition, state checkpoints, progress estimation, plan repair, and replanning. AgentQuest illustrates the value of tracking intermediate progress and identifying failure points within agent trajectories (Gioacchini et al., 2024).

Long-horizon comparisons should measure intermediate progress, goal coherence, cumulative constraint violations, checkpoint recoverability and unnecessary actions. Controlled experiments can replace one planner or state-management mechanism at a time under matched environment, tool and model configurations. Such designs offer more causal insight than comparing unmatched benchmark leaderboards.

9.5 Persistent and Continual Memory

Future research should treat memory as a managed knowledge lifecycle rather than a growing repository. Architectural policies are needed for retention, consolidation, retrieval, contradiction resolution, provenance, forgetting, and access control. StreamBench evaluates whether language agents improve over continuing input-feedback sequences rather than only performing well at a fixed point in time (Wu et al., 2024).

Memory systems should be tested with contradictory updates, stale observations, multi-session tasks and controlled retrieval contamination. Evaluations must distinguish storage volume, retrieval precision and usefulness to downstream actions. Shared memory requires additional metrics for conflicting updates and unauthorized access.

9.6 Hybrid Neuro-Symbolic Agentic Systems

The central research problem is not simply combining neural and symbolic approaches, but determining which architectural responsibilities should be neural and which should be symbolic. Hakim et al. (2026) identify integration patterns and interface design as central open questions for neuro-symbolic Agentic AI. Future architectures require reliable translation between natural-language representations and formal structures.

Neuro-symbolic research should focus on the accuracy of interfaces between representations. Translating a natural-language objective into formal operators can introduce errors that formal planners cannot detect. Studies should measure translation validity and downstream constraint satisfaction alongside end-to-end success and integration costs.

9.7 Robust Multi-Agent Coordination

Multi-agent systems require stronger coordination mechanisms as the number and heterogeneity of agents increase. Research on distributed online meta-learning demonstrates one approach to adaptive collaboration among multiple agents (Luo et al., 2025). ACNBP introduces mechanisms for capability discovery, negotiation, and binding among heterogeneous agents (Huang et al., 2025b). Future work should examine coordination contracts, shared-state consistency, conflict resolution, deadlock detection, adaptive topology, and system-level monitoring.

Coordination should be tested through controlled changes in topology, message volume, agent specialization and state-sharing policy. Studies should hold underlying models and tools constant where feasible. The key question is when parallelism and differentiated information access outweigh the cost and risk of distributed decision-making.

9.8 Reflection, Verification, and Failure Recovery

Reflection does not guarantee reliable correction. Reflection-Bench confirms that contemporary models continue to show limitations in meta-reflection and related epistemic capabilities (Li et

al., 2025b). Future architectures should therefore separate reflection, verification, and recovery. A mature recovery loop should support detection, diagnosis, repair, verification, and resumption.

Useful recovery experiments should inject known faults into information retrieval, memory updates, plans, tools and communications. Investigators can then evaluate detection, diagnosis, corrective action and verification separately. This avoids treating any rewritten answer as evidence of successful error correction.

9.9 Interoperable Tools and Agent Communication Protocols

Future Agentic AI ecosystems are likely to combine agents, tools, models, memory systems, and external services developed by different organizations. A2A addresses agent communication, while MCP provides structured connections between models and external tools. Jeong (2025) examines their combined use for interoperability among LLM-based autonomous agents. Protocols must also support capability discovery and negotiation.

Protocol research must preserve semantic meaning, capabilities and delegated authority across framework boundaries. Shared tool schemas are insufficient where participants disagree about task completion, permissible actions or failed execution. Capability descriptions should document inputs, outputs, permissions, constraints and evidence supporting advertised functions.

9.10 Evaluation of Architecture-Level Agency

Agent evaluation still relies heavily on task success. Future evaluation should operate at capability, trajectory, and architecture levels. Cihon et al. (2025) propose measuring agent autonomy through properties of orchestration code, while AgentQuest provides complementary trajectory-level metrics for tracking progress and locating failures (Gioacchini et al., 2024). Surveys of agent evaluation similarly highlight the need to move beyond narrow end-task performance (Yehudai et al., 2025).

Architecture-level evaluation must distinguish model effects from planning, memory, execution and organization effects. Studies should report agent configurations, tools, task environments and compute budgets, and use repeated trials for stochastic behaviors. Coding the capability-realization instance before benchmarking could help identify meaningful intervention points.

9.11 Bounded, Explainable, and Auditable Autonomy

Research should avoid treating maximum autonomy as the endpoint of Agentic AI development. Different actions require different levels of delegated authority. Future architectures should support dynamic autonomy boundaries. Low-impact and reversible actions may proceed automatically, while high-consequence or uncertain actions may require stronger verification or human authorization. Agent characterization research already represents autonomy as a graded rather than binary property (Kasirzadeh & Gabriel, 2025).

The objective of research should be appropriate bounded autonomy rather than maximum unattended action. Evaluation can examine authorization violations, time to successful intervention, operator understanding of proposed actions and failures occurring after delayed approval. Governance must remain specific to the deployment context.

9.12 Computational Efficiency and Sustainable Agentic AI

Agentic systems can require many model calls for planning, retrieval, critique, tool interaction, and multi-agent communication. Efficiency research should move toward adaptive resource allocation. Systems should determine when a large model is necessary, when smaller models are sufficient, when reflection adds value, and when additional agent communication is unnecessary. Peykani et al. (2026b) examine metrics and mitigation strategies for Green AI, while Cheung et al. (2025) investigate carbon effects associated with LLM-assisted development.

Efficiency should be expressed relative to verified task success, rather than simply token counts or agent numbers. Routing small tasks to lighter models, selective retrieval and adaptive verification are candidate strategies requiring measured comparison. Environmental assessments need defensible workload and energy assumptions.

9.13 Toward Standardized and Empirically Evaluated Reference Architectures

The preceding research directions converge on the need for reference architectures that provide a stable functional vocabulary without imposing one implementation technology. A useful reference architecture should identify common functional interfaces while allowing symbolic, neural, hybrid, and multi-agent realizations. White (2024) argues for software systems that combine generative and agentic mechanisms as evolving computational structures. The reference architecture proposed in Section 6 should therefore be treated as a testable synthesis rather than a final standard.

Table 16. Prioritized research roadmap for Agentic AI architectures.

Research direction	Priority phase	Next step	Evidence of progress
Conceptual foundations	I	Operational definitions and capability boundaries	Classification agreement
Architecture reporting	I–II	Common function-mechanism specification	Reporting completeness
Formal models	II	Explicit state/control constraints	Verification under defined assumptions
Long-horizon reliability	III	State checkpoints and plan repair	Trajectory recovery studies
Persistent memory	III	Provenance and contradiction resolution	LongMemEval-style testing
Neuro-symbolic integration	II–III	Validated representation translation	Constraint and translation tests
Multi-agent coordination	IV	Contracts and shared-state protocols	Conflict and overhead measures
Recovery and reflection	III	Diagnosis and independent verification	Controlled fault injection
Tool interoperability	IV	Semantic protocols and capability descriptions	Cross-framework tests
Architectural evaluation	III–V	Matched architecture ablations	Repeatable comparative evidence
Bounded autonomy	V	Authority and human-oversight controls	Authorization and intervention tests
Sustainable execution	V	Adaptive compute allocation	Energy/cost per successful task
Reference architecture	II–V	Independent taxonomy validation	Coverage and discriminative usefulness

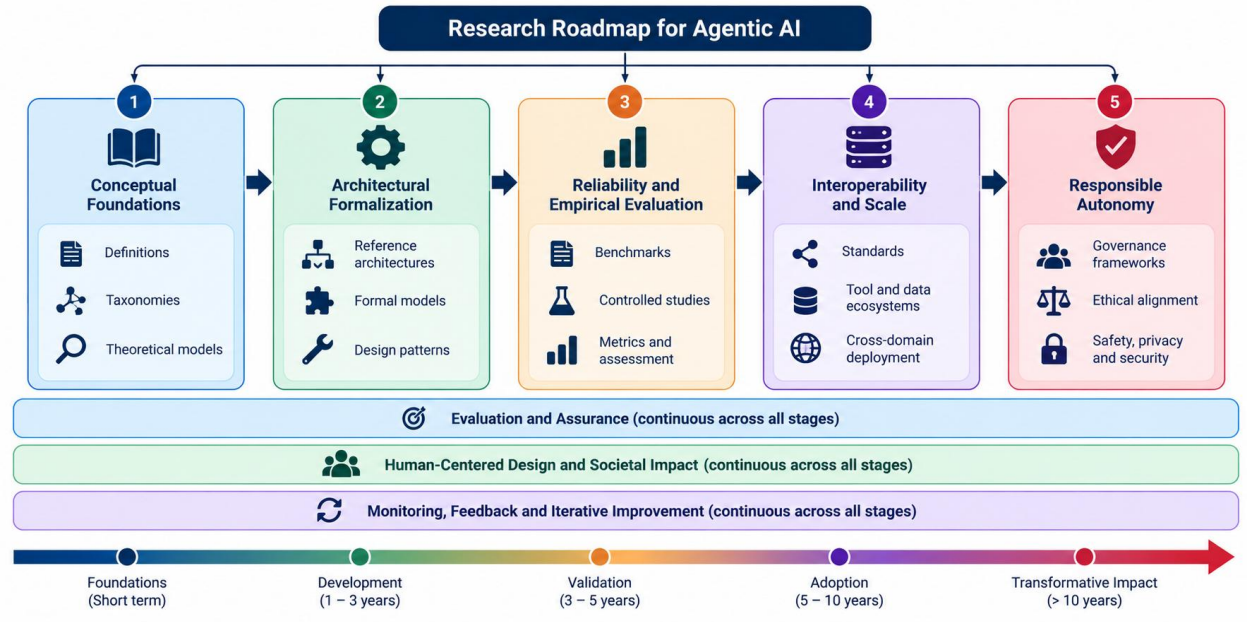


Fig. 7. Prioritized research roadmap for Agentic AI: from conceptual foundations to reliable, interoperable, and responsible architectures. Conceptual visualization prepared with generative-AI assistance; design and scientific content require author verification.

The proposed reference architecture is a testable analytical construct, not a mature standard. Future validation should examine coding agreement, coverage of unconventional agent designs, the ability to discriminate mechanisms that share a capability label, and whether the reference structure improves technical documentation or comparative evaluation. Continuous assurance and empirical testing should span every phase of the roadmap, as Fig. 7 emphasizes.

10. Limitations of the Survey

This comprehensive survey uses a concept-centered, non-systematic literature collection process. Consequently, it does not establish that all eligible studies were identified, and the distribution of papers across paradigms cannot be interpreted as field-wide prevalence. The document does not provide a preregistered protocol, independent screening decisions, bibliometric coverage estimates, or a formal risk-of-bias appraisal. Selection of representative mechanisms may privilege well-documented systems and English-language, accessible publications.

The conceptual boundaries of Agentic AI remain contested. The nine capabilities describe recurring functional categories, not necessary-and-sufficient conditions for all possible systems.

Multi-agent organization intersects with, rather than excludes, symbolic, neural, and hybrid mechanisms. Any proposed separation of functional roles is an analytical convenience: practical implementations often merge roles within one model or distribute a role across several services.

The taxonomy, mapping schema, formal abstraction, and reference architecture have not been validated through experiments, expert consensus, independent coding agreement, formal verification, or comparative deployment. Mechanism associations and responsible-design considerations are explanatory synthesis claims; they are not numerical performance findings or certification evidence. Future work should operationalize coding rules, test inter-rater reliability, construct architectural case studies, and compare implementations under matched tasks, budgets, and safety conditions.

The reference set is current to the reported October 2026 cut-off, subject to verification of individual bibliographic records. Fast-moving preprint work may be superseded, and the contribution should be rechecked against new taxonomies and architectures before journal submission.

The proposed capability-realization schema, its classification rules, formal abstraction and reference architecture have not been validated through new benchmark experiments, expert consensus, inter-rater studies or controlled deployments. Literature sources have different levels of reporting completeness and were not selected through an exhaustive review protocol. Bibliographic records also require final verification against their publication versions. This survey therefore does not provide evidence of empirical superiority or regulatory compliance.

11. Conclusion

This survey examined Agentic Artificial Intelligence from an architecture-centered perspective. It clarified the conceptual boundaries of Agentic AI, synthesized its recurring capabilities, compared the principal architectural paradigms through which those capabilities are realized, and integrated the findings into a unified architecture-capability taxonomy and reference architecture. The analysis shows that Agentic AI is more appropriately understood as a systems paradigm than as a particular model, framework, or implementation technique.

11.1 Summary of Key Findings

With respect to RQ1, the survey finds that Agentic AI is conceptually distinct from traditional AI, Generative AI, individual AI agents, autonomous systems, and multi-agent systems. Its defining characteristic is not a single technology but the integration of goal-directed, stateful, and feedback-sensitive capabilities that support sustained interaction with an environment. Multi-agent organization can strengthen or distribute agency, but it is not a necessary condition for a system to be agentic.

With respect to RQ2, the literature converges around a recurring capability structure comprising goal and context management, perception and state representation, reasoning and decision-making, planning and replanning, memory and knowledge, action and tool use, reflection and evaluation, learning and adaptation, and communication and coordination where distributed agency is present. Long-horizon persistence and autonomy emerge from the sustained interaction of these capabilities rather than from any isolated component.

With respect to RQ3, symbolic/classical, neural/generative, hybrid neuro-symbolic, and multi-agent architectures realize these capabilities through different mechanisms. No single architectural paradigm provides a universal solution across all task and system requirements.

With respect to RQ4, the synthesis demonstrates that recurring agentic capabilities can be separated conceptually from their realization mechanisms and organized within a common reference structure. Explainability, human oversight, safety, accountability, privacy, and security operate across this structure as architectural assurance concerns.

11.2 Scholarly Contributions and Implications

The first contribution is an operational definition of Agentic AI that places agency at the system level. The second is a core capability taxonomy that separates recurring functions from implementation mechanisms. The third is an architectural paradigm taxonomy covering symbolic/classical, neural/generative, hybrid neuro-symbolic, and multi-agent approaches. The fourth is the architecture-capability realization matrix, which demonstrates that the same functional capability can be implemented through substantially different mechanisms. The fifth is the integrated architecture-capability taxonomy and unified reference architecture, which provide a paradigm-neutral structure for describing how goals, state, reasoning, planning, memory, action, feedback, adaptation, and coordination interact to produce agentic behavior.

These contributions position Agentic AI as an integrated systems concept rather than a synonym for autonomous language models. The synthesis also reinforces the importance of studying relationships among components rather than capabilities in isolation. Agentic system quality depends on how capabilities are integrated, governed, and sustained across time.

The most distinctive contribution is the capability-realization instance: a unit of comparison specifying what a function does, which mechanism implements it, where responsibility resides, which state and action interfaces are required, and what evidence supports the assignment. It supports transparent comparisons across symbolic, neural, hybrid and distributed systems without claiming that different implementations have equivalent reliability. Its reproducibility, discriminative usefulness and practical value remain appropriate subjects for independent research.

11.3 Final Perspective

Agentic AI marks a shift from isolated model inference toward systems capable of maintaining goals, reasoning over changing state, planning actions, using memory and tools, evaluating outcomes, adapting behavior, and coordinating distributed capabilities. Its central research challenge is consequently not only how to make individual models more capable, but how to organize these capabilities into coherent architectures that remain reliable, interpretable, controllable, interoperable, and sustainable over extended interactions. By separating capability from realization mechanism and integrating recurring functions into a unified taxonomy and reference architecture, this survey provides a foundation for studying Agentic AI as an architectural discipline rather than as a collection of rapidly changing models, agents, and frameworks.

Abbreviations

Abbreviation	Full form
A2A	Agent2Agent
ACT-R	Adaptive Control of Thought—Rational
ACM	Association for Computing Machinery
AI	Artificial Intelligence
AI RMF	Artificial Intelligence Risk Management Framework

API	Application Programming Interface
BDI	Belief–Desire–Intention
CoT	Chain-of-Thought
EU	European Union
GAIA	General AI Assistants
GUI	Graphical User Interface
HTN	Hierarchical Task Network
IEEE	Institute of Electrical and Electronics Engineers
ISO/IEC	International Organization for Standardization / International Electrotechnical Commission
LLM	Large Language Model
LLM+P	Large Language Models + Planning
MAS	Multi-Agent Systems
MCP	Model Context Protocol
NIST	National Institute of Standards and Technology
ORCID	Open Researcher and Contributor ID
OWASP	Open Worldwide Application Security Project
PDDL	Planning Domain Definition Language
RAG	Retrieval-Augmented Generation
RL	Reinforcement Learning
RT-2	Robotics Transformer 2
STRIPS	Stanford Research Institute Problem Solver
SWE-bench	Software Engineering Benchmark
TB-CSPN	Topic-Based Communication Space Petri Net
ToT	Tree of Thoughts

Statements and Declarations

Author Contributions

All ten authors contributed to the development, preparation, and refinement of this comprehensive survey. Ibrahim Adabara led the conceptualization, development of the review framework, architecture-capability synthesis, and preparation of the original manuscript.

Abigaba Sofia, Kyampaire Sarah, Rumanzi Smith, and Kisembo Moreen contributed to literature assessment, research material organization, and manuscript review. Yale Ibrahim Danjuma, Elhadi Abdalla A. Arbab, and Venkateswarlu Maninti contributed to methodological and technical evaluation and critical refinement. Kanyesigye Simon and Ninyesiga Allan contributed to technical review and manuscript refinement. This proposed contribution allocation must be confirmed by each author before submission; all authors must approve the final manuscript.

Funding

The authors received no external financial support for the research, authorship, or publication of this article.

Competing Interests

The authors declare that they have no competing interests. This statement must be confirmed by all authors before submission.

Ethics Approval and Consent to Participate

Not applicable. This comprehensive narrative survey synthesizes previously published literature and reports no new experiments involving human participants or animals.

Consent for Publication

Not applicable. The manuscript does not contain identifiable individual participant information requiring consent for publication.

Data Availability

No new primary datasets were generated or analyzed for the current narrative review. The literature supporting the synthesis is identified in the reference list; a full reproducible study-screening dataset was not compiled.

Declaration of Generative AI Assistance

Generative AI (OpenAI ChatGPT and its image-generation functionality) assisted with structural outlining, manuscript drafting and editing, bibliographic organization, and the design of conceptual illustrations (Figs. 1–7). The diagrams synthesize author-specified conceptual relationships and are not original experimental measurements. All scientific content, source

attribution, conceptual claims, and figure accuracy require independent author review and verification. The authors retain full accountability for the final submitted work.

References

Abou Ali, M., Dornaika, F., & Charafeddine, J. (2026). Agentic AI: A comprehensive survey of architectures, applications, and future directions. Artificial Intelligence Review, 59, 11. doi:10.1007/s10462-025-11422-4

Acharya, D. B., Kuppan, K., & Divya, B. (2025). Agentic AI: Autonomous intelligence for complex goals: A comprehensive survey. IEEE Access, 13, 18912-18936. doi:10.1109/ACCESS.2025.3532853

Ahn, M., Brohan, A., Brown, N., Chebotar, Y., Cortes, O., David, B., et al. (2022). Do as I can, not as I say: Grounding language in robotic affordances. arXiv. doi:10.48550/arXiv.2204.01691

Ala'anzy, M. A., Oshayeva, M., & Himeur, Y. (2026). Autonomous agents and multi-agent systems: A comprehensive review. Artificial Intelligence Review. doi:10.1007/s10462-026-11731-2

Anderson, J. R., Matessa, M., & Lebiere, C. (1997). ACT-R: A theory of higher level cognition and its relation to visual attention. Human-Computer Interaction, 12(4), 439-462. doi:10.1207/s15327051hci1204_5

Arunkumar, V., Gangadharan, G. R., & Buyya, R. (2026). Agentic artificial intelligence (AI): Architectures, taxonomies, and evaluation of large language model agents. arXiv. doi:10.48550/arXiv.2601.12560

Asai, A., Wu, Z., Wang, Y., Sil, A., & Hajishirzi, H. (2024). Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In Proceedings of the International Conference on Learning Representations (ICLR). https://openreview.net/forum?id=hSyW5go0v8

Bai, Y., Kadavath, S., Kundu, S., Askell, A., Kernion, J., Jones, A., et al. (2022). Constitutional AI: Harmlessness from AI feedback. arXiv. doi:10.48550/arXiv.2212.08073

Bajoria, S., Ranjan, S., Adhitya, M., Suri, V., Hassija, V., Chamola, V., & Hussain, A. (2026). From language models to agentic AI: A survey of autonomous, action-enabled, and collaborative LLM agents. Cognitive Computation, 18, Article 103. doi:10.1007/s12559-026-10619-1

Bandi, A., Kongari, B., Naguru, R., Pasnoor, S., & Vilipala, S. V. (2025). The rise of Agentic AI: A review of architectures, applications, and challenges. Future Internet, 17, 404. doi:10.3390/fi17090404

Bansod, P. B. (2025). Distinguishing autonomous AI agents from collaborative agentic systems: A comprehensive framework for understanding modern intelligent architectures. arXiv. doi:10.48550/arXiv.2506.01438

Belcak, P., Heinrich, G., Diao, S., Fu, Y., Dong, X., Muralidharan, S., et al. (2025). Small language models are the future of agentic AI. arXiv. doi:10.48550/arXiv.2506.02153

Bidochko, A., & Vyklyuk, Y. (2026). Thought management system for long-horizon, goal-driven LLM agents. Journal of Computational Science, 93, 102740. doi:10.1016/j.jocs.2025.102740

Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., et al. (2021). On the opportunities and risks of foundation models. arXiv. doi:10.48550/arXiv.2108.07258

Borghoff, U. M., Bottoni, P., & Pareschi, R. (2025). Beyond prompt chaining: Structured control for agentic workflows using colored Petri nets. Future Internet, 17(8), 363. doi:10.3390/fi17080363

Botti, V. (2025). Agentic AI and multiagentic: Are we reinventing the wheel? arXiv. doi:10.48550/arXiv.2506.01463

Bran, A. M., Cox, S., Schilter, O., Baldassari, C., White, A. D., & Schwaller, P. (2024). Augmenting large language models with chemistry tools. Nature Machine Intelligence, 6(5), 525–535. doi:10.1038/s42256-024-00832-8

Brohan, A., Brown, N., Carbajal, J., Chebotar, Y., Dabis, J., Finn, C., et al. (2023). RT-2: Vision-language-action models transfer web knowledge to robotic control. arXiv. doi:10.48550/arXiv.2307.15818

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., et al. (2020). Language models are few-shot learners. Advances in Neural Information Processing Systems, 33, 1877-1901. doi:10.48550/arXiv.2005.14165

Chen, W., Su, Y., Zuo, J., Yang, C., Yuan, C., Qian, C., & Zhou, J. (2023). AgentVerse: Facilitating multi-agent collaboration and exploring emergent behaviors in agents. arXiv. doi:10.48550/arXiv.2308.10848

Cheng, Y., Zhang, C., Zhang, Z., Meng, X., Hong, S., Li, W., et al. (2024). Exploring large language model based intelligent agents: Definitions, methods, and prospects. arXiv. doi:10.48550/arXiv.2401.03428

Cheung, K. S., Kaul, M., Jahangirova, G., Mousavi, M. R., & Zie, E. (2025). Comparative analysis of carbon footprint in manual vs. LLM-assisted code development. In Proceedings of the 1st International Workshop on Responsible Software Engineering (pp. 13-20). doi:10.1145/3711919.3728678

Cihon, P., Stein, G., Bansal, H., Manning, C. D., & Xu, K. (2025). Measuring AI agent autonomy: Towards a scalable approach with code inspection. arXiv. doi:10.48550/arXiv.2502.15212

de Silva, L., Meneguzzi, F., & Logan, B. (2020). BDI agent architectures: A survey. In Proceedings of the 29th International Joint Conference on Artificial Intelligence (pp. 4914-4921). doi:10.24963/ijcai.2020/684

Debenedetti, E., Zhang, J., Balunović, M., Beurer-Kellner, L., Fischer, M., & Tramèr, F. (2024). AgentDojo: A dynamic environment to evaluate prompt injection attacks and defenses for LLM agents. In Advances in Neural Information Processing Systems, 37. doi:10.52202/079017-2636

Derouiche, H., Brahmi, Z., & Mazeni, H. (2025). Agentic AI frameworks: Architectures, protocols, and design challenges. arXiv. doi:10.48550/arXiv.2508.10146

Doshi-Velez, F., & Kortz, M. (2017). Accountability of AI under the law: The role of explanation. Berkman Klein Center. https://cyber.harvard.edu/publications/2017/11/AI-explainability

Driess, D., Xia, F., Sajjadi, M. S. M., Lynch, C., Chowdhery, A., Ichter, B., et al. (2023). PaLM-E: An embodied multimodal language model. arXiv. doi:10.48550/arXiv.2303.03378

Du, Y., Li, S., Torralba, A., Tenenbaum, J. B., & Mordatch, I. (2024). Improving factuality and reasoning in language models through multiagent debate. In Proceedings of the International Conference on Machine Learning (ICML). https://arxiv.org/abs/2305.14325

Durante, Z., Sarkar, S., et al. (2024). Agent AI: Surveying the horizons of multimodal interaction. arXiv. doi:10.48550/arXiv.2401.03568

Erisken, S., Gothard, T., Leitgab, M., & Potham, R. (2025). MAEBE: Multi-agent emergent behavior framework. arXiv. doi:10.48550/arXiv.2506.03053

European Parliament & Council of the European Union. (2024). Regulation (EU) 2024/1689 laying down harmonised rules on artificial intelligence (Artificial Intelligence Act). Official Journal of the European Union. http://data.europa.eu/eli/reg/2024/1689/oj

Feng, X., et al. (2025). Levels of autonomy for AI agents. arXiv. doi:10.48550/arXiv.2502.01635

Feuerriegel, S., Hartmann, J., Janiesch, C., & Zschech, P. (2024). Generative AI. Business & Information Systems Engineering, 66, 111-126. doi:10.1007/s12599-023-00834-7

Fikes, R. E., & Nilsson, N. J. (1971). STRIPS: A new approach to the application of theorem proving to problem solving. Artificial Intelligence, 2(3-4), 189-208. doi:10.1016/0004-3702(71)90010-5

Gioacchini, L., et al. (2024). AgentQuest: A modular benchmark framework to measure progress and improve LLM agents. In Proceedings of NAACL 2024: System Demonstrations. doi:10.18653/v1/2024.naacl-demo.19

Gohil, V., Shah, S. B., Rauniyar, K., Shiwakoti, S., Thapa, S., Sheng, D., ... Naseem, U. (2026). Towards safe and trustworthy agentic AI: Foundations, taxonomy, technologies, applications, and future directions. Artificial Intelligence Review. doi:10.1007/s10462-026-11682-8

Gou, Z., Shao, Z., Gong, Y., Shen, Y., Yang, Y., Huang, M., et al. (2024). CRITIC: Large language models can self-correct with tool-interactive critiquing. International Conference

on Learning Representations. https://proceedings.iclr.cc/paper_files/paper/2024/hash/fef126561bbf9d4467dbb8d27334b8fe-Abstract-Conference.html

Greshake, K., Abdelnabi, S., Mishra, S., Endres, C., Holz, T., & Fritz, M. (2023). Not what you've signed up for: Compromising real-world LLM-integrated applications with indirect prompt injection. arXiv preprint arXiv:2302.12173. doi:10.48550/arXiv.2302.12173

Guo, Z., et al. (2024). StableToolBench: Towards stable large-scale benchmarking on tool learning of large language models. arXiv. doi:10.48550/arXiv.2403.07714

Habler, E., et al. (2025). Secure agentic AI applications using agent-to-agent communication protocols. arXiv. doi:10.48550/arXiv.2504.16902

Hakim, S. B., Adil, M., Velasquez, A., & Song, H. H. (2026). Neuro-symbolic agentic AI: Architectures, integration patterns, applications, open challenges and future research directions. Computer Science Review, 60, 100902. doi:10.1016/j.cosrev.2026.100902

Hammond, L., Chan, A., Clifton, J., Hoelscher-Obermaier, J., Khan, A., McLean, E., & Rahwan, I. (2025). Multi-agent risks from advanced AI. arXiv. doi:10.48550/arXiv.2502.14143

Hong, S., Zheng, X., Chen, J., Cheng, Y., Zhang, C., Wang, Z., et al. (2024). MetaGPT: Meta programming for multi-agent collaborative framework. International Conference on Learning Representations. https://openreview.net/forum?id=VtmBAGCN7o

Hua, W., et al. (2024). TrustAgent: Towards safe and trustworthy LLM-based agents. Findings of EMNLP 2024. https://aclanthology.org/2024.findings-emnlp.585/

Huang, C., et al. (2025b). Agent capability negotiation and binding protocol for heterogeneous agent ecosystems. arXiv. doi:10.48550/arXiv.2506.13590

Huang, K., Yin, H., Huang, H., & Gao, W. (2023). Towards Green AI in fine-tuning large language models via adaptive backpropagation. arXiv. doi:10.48550/arXiv.2309.13192

Huang, L., et al. (2025a). A survey on hallucination in large language models. ACM Transactions on Intelligent Systems and Technology. doi:10.1145/3703155

Hughes, L., et al. (2025). Agentic AI and autonomous intelligent systems. Journal of Computer Information Systems. doi:10.1080/08874417.2025.2483832

International Organization for Standardization & International Electrotechnical Commission. (2023). Information technology—Artificial intelligence—Management system (ISO/IEC Standard No. 42001:2023). https://www.iso.org/standard/42001

Jennings, N. R., Sycara, K., & Wooldridge, M. (1998). A roadmap of agent research and development. Autonomous Agents and Multi-Agent Systems, 1(1), 7-38. doi:10.1023/A:1010090405266

Jeong, J. (2025). A study on the MCP x A2A framework for enhancing interoperability of LLM-based autonomous agents. arXiv. doi:10.48550/arXiv.2506.01804

Jiang, P., Sonne, C., Li, W., You, F., & You, S. (2024). Preventing the immense increase in the life-cycle energy and carbon footprints of LLM-powered intelligent chatbots. Engineering, 40, 202-210. doi:10.1016/j.eng.2024.04.002

Jimenez, C. E., Yang, J., Wettig, A., Yao, S., Pei, K., Press, O., & Narasimhan, K. R. (2024). SWE-bench: Can language models resolve real-world GitHub issues? In Proceedings of the International Conference on Learning Representations (ICLR). https://openreview.net/forum?id=VTF8yNQM66

Joshi, H., Liu, S., Chen, J., Weigle, L., & Lam, M. (2025). Controllable and reliable knowledge-intensive task-oriented conversational agents with declarative Genie worksheets. In Proceedings of ACL 2025. https://aclanthology.org/2025.acl-long.1323/

Kaelbling, L. P., Littman, M. L., & Cassandra, A. R. (1998). Planning and acting in partially observable stochastic domains. Artificial Intelligence, 101(1-2), 99-134. doi:10.1016/S0004-3702(98)00023-X

Kasirzadeh, A., & Gabriel, I. (2025). Characterizing AI agents for alignment and governance. arXiv. doi:10.48550/arXiv.2504.21848

Khapre, S., Mersha, M. A., Olalekan, Z., Shakil, H., Iskander, S., Moustafa, A., & Kalita, J. (2026). Agentic AI systems: A systematic survey of multi-agent architectures, cognitive

foundations, interaction, explainability, security, and performance evaluation. Neurocomputing, 696, Article 134049. doi:10.1016/j.neucom.2026.134049

Laird, J. E., Newell, A., & Rosenbloom, P. S. (1987). SOAR: An architecture for general intelligence. Artificial Intelligence, 33(1), 1–64. doi:10.1016/0004-3702(87)90050-6

Lee, D., & Tiwari, M. (2024). Prompt infection: LLM-to-LLM prompt injection within multi-agent systems. arXiv. doi:10.48550/arXiv.2410.07283

Li, C., et al. (2025b). Reflection-Bench: Probing AI intelligence with reflection. Proceedings of Machine Learning Research, 267. https://proceedings.mlr.press/v267/li25cu.html

Li, G., Hammoud, H. A. A. K., Itani, H., Khizbullin, D., & Ghanem, B. (2023). CAMEL: Communicative agents for mind exploration of large scale language model society. arXiv. doi:10.48550/arXiv.2303.17760

Li, H., et al. (2024). Formal-LLM: Integrating formal language and natural language for controllable agents. arXiv. doi:10.48550/arXiv.2402.00798

Li, W., Lin, J., Jiang, Z., Cao, J., Liu, X., Zhang, J., & Zhou, W. (2025a). Chain-of-agents: End-to-end agent foundation models via multi-agent distillation and agentic RL. arXiv. doi:10.48550/arXiv.2508.13167

Li, Y., et al. (2026). Agentic AI systems: Capabilities, architectures, and organizational implications. Business & Information Systems Engineering. doi:10.1007/s12599-026-01009-w

Liu, B., et al. (2023a). LLM+P: Empowering large language models with optimal planning proficiency. arXiv. doi:10.48550/arXiv.2304.11477

Liu, V., & Yin, Y. (2024). Green AI: Exploring carbon footprints, mitigation strategies, and trade offs in large language model training. Discover Artificial Intelligence, 4(1), 49. doi:10.1007/s44163-024-00149-w

Liu, X., Yu, H., Zhang, H., Xu, Y., Lei, X., Lai, H., ... Tang, J. (2024). AgentBench: Evaluating LLMs as agents. In Proceedings of the International Conference on Learning Representations (ICLR).

https://proceedings.iclr.cc/paper_files/paper/2024/hash/e9df36b21ff4ee211a8b71ee8b7e9f57-Abstract-Conference.html

Liu, Z., Yao, W., Zhang, J., Yang, H., & others. (2024). AgentLite: A lightweight library for building and advancing task-oriented LLM agent systems. arXiv. doi:10.48550/arXiv.2402.15538

Luo, Z., et al. (2025). Distributed online meta-learning for multi-agent collaboration and adaptation. Journal of Industrial Information Integration, 100852. doi:10.1016/j.jii.2025.100852

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. arXiv. doi:10.48550/arXiv.2507.06850

Madaan, A., Tandon, N., Gupta, P., Hallinan, S., Gao, L., Wiegreffe, S., et al. (2023). Self-Refine: Iterative refinement with self-feedback. arXiv. doi:10.48550/arXiv.2303.17651

Masoor, H. (2025). SAMEP: A secure protocol for persistent context sharing across AI agents. arXiv. doi:10.48550/arXiv.2507.10562

Masterman, T., Besen, S., Sawtell, M., & Chao, A. (2024). The landscape of emerging AI agent architectures for reasoning, planning, and tool calling: A survey. arXiv. doi:10.48550/arXiv.2404.11584

McDermott, D., Ghallab, M., Howe, A., Knoblock, C., Ram, A., Veloso, M., ... Wilkins, D. (1998). PDDL—The planning domain definition language (Technical Report CVC TR-98-003/DCS TR-1165). Yale Center for Computational Vision and Control. https://www.cs.yale.edu/homes/dvm/papers/pddl.pdf

Mialon, G., Fourrier, C., Wolf, T., LeCun, Y., & Scialom, T. (2024). GAIA: A benchmark for general AI assistants. In Proceedings of the International Conference on Learning Representations (ICLR). https://proceedings.iclr.cc/paper_files/paper/2024/hash/25ae35b5b1738d80f1f03a8713e405ec-Abstract-Conference.html

Miehling, E., Ramamurthy, K. N., Varshney, K. R., & Riemer, M. (2025). Agentic AI needs a systems theory. arXiv. doi:10.48550/arXiv.2503.00237

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., et al. (2015). Human-level control through deep reinforcement learning. Nature, 518, 529-533. doi:10.1038/nature14236

Nageshwaran, V., Ezekiel, S., Tran, T. T., & Narasimhan, V. L. (2026). Large language model agent evaluation and benchmarking: A systematic survey, meta-taxonomy, and critical research roadmap. Artificial Intelligence Review. doi:10.1007/s10462-026-11678-4

Newell, A., & Simon, H. A. (1976). Computer science as empirical inquiry: Symbols and search. Communications of the ACM, 19(3), 113-126. doi:10.1145/360018.360022

Open Worldwide Application Security Project. (2025). OWASP Top 10 for agentic applications for 2026. https://genai.owasp.org/resource/owasp-top-10-for-agentic-applications-for-2026/

Packer, C., Fang, V., Patil, S. G., Lin, K., Wooders, S., & Gonzalez, J. E. (2023). MemGPT: Towards LLMs as operating systems. arXiv. doi:10.48550/arXiv.2310.08560

Park, J. S., O'Brien, J., Cai, C. J., Morris, M. R., Liang, P., & Bernstein, M. S. (2023). Generative agents: Interactive simulacra of human behavior. In Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology. doi:10.1145/3586183.3606763

Patil, S. G., Zhang, T., Wang, X., & Gonzalez, J. E. (2023). Gorilla: Large language model connected with massive APIs. arXiv. doi:10.48550/arXiv.2305.15334

Peykani, P., Emrouznejad, A., Ghanidel, S., Javadi-Sisi, I., & Mirjalili, S. (2026a). Green artificial intelligence: A comprehensive review of metrics, tools, challenges, trends, and future prospects. Archives of Computational Methods in Engineering. doi:10.1007/s11831-026-10546-2

Peykani, P., Ghanidel, S., Javadi-Sisi, I., Snasel, V., & Mirjalili, S. (2026b). A holistic review of Agentic AI frameworks, applications, and research trajectories. Archives of Computational Methods in Engineering. doi:10.1007/s11831-026-10675-8

Qin, Y., et al. (2025). Tool learning with foundation models. ACM Computing Surveys. doi:10.1145/3704435

Qin, Y., Liang, S., Ye, Y., Zhu, K., Yan, L., Lu, Y., et al. (2023). ToolLLM: Facilitating large language models to master 16000+ real-world APIs. arXiv. doi:10.48550/arXiv.2307.16789

Raghavendra, S., & Saikia, M. J. (2026). Agentic AI: A perspective on architecture, frameworks and applications. AI, 7, 219. doi:10.3390/ai7060219

Rao, A. S., & Georgeff, M. P. (1995). BDI agents: From theory to practice. In Proceedings of the First International Conference on Multi-Agent Systems (pp. 312–319). MIT Press. https://apice.unibo.it/bin/view/Publication/BdiIcmas95

Rebedea, T., Dinu, R., Sreedhar, M., Parisien, C., & Cohen, J. (2023). NeMo Guardrails: A toolkit for controllable and safe LLM applications with programmable rails. In Proceedings of EMNLP 2023: System Demonstrations. https://aclanthology.org/2023.emnlp-demo.40/

Saadi, A., Maamri, R., & Sahnoun, Z. (2020). Behavioral flexibility in belief-desire-intention architectures. Multiagent and Grid Systems, 16(4), 343-377. doi:10.3233/MGS-200335

Santoni de Sio, F., & van den Hoven, J. (2018). Meaningful human control over autonomous systems: A philosophical account. Frontiers in Robotics and AI, 5, 15. doi:10.3389/frobt.2018.00015

Sapkota, R., Roumeliotis, K. I., & Karkee, M. (2026). AI agents vs. Agentic AI: A conceptual taxonomy, applications and challenges. Information Fusion, 126, 103599. doi:10.1016/j.inffus.2025.103599

Schick, T., Dwivedi-Yu, J., Dessi, R., Raileanu, R., Lomeli, M., Hambro, E., et al. (2023). Toolformer: Language models can teach themselves to use tools. arXiv. doi:10.48550/arXiv.2302.04761

Schneider, J. (2025). Generative to Agentic AI: Survey, conceptualization, and challenges. arXiv. doi:10.48550/arXiv.2504.18875

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

Processing Systems, 36, 8634-8652. https://proceedings.neurips.cc/paper_files/paper/2023/hash/1b44b878bb782e6954cd888628510e90-Abstract-Conference.html

Silver, T., Hariprasad, V., Shuttleworth, R. S., Kumar, N., Lozano-Perez, T., & Kaelbling, L. P. (2022). PDDL planning with pretrained large language models. https://drive.google.com/file/d/1PqCYzzfdUFitxG7NxFs0D6dn-TzI97q_/view

Singh, A., et al. (2024). Agentic workflows for large language model systems. In IEEE AIIoT 2024. doi:10.1109/AIIoT61789.2024.10578990

Singh, A., et al. (2025). Agentic retrieval-augmented generation: A survey. arXiv. doi:10.48550/arXiv.2501.09136

Stoica, I., et al. (2024). From models to compound AI systems: The importance of system specifications. arXiv. doi:10.48550/arXiv.2412.05299

Stone, P., & Veloso, M. (2000). Multiagent systems: A survey from a machine learning perspective. Autonomous Robots, 8, 345-383. doi:10.1023/A:1008942012299

Subramanian, J., et al. (2024). Neuro-symbolic multi-agent reinforcement learning. arXiv. doi:10.48550/arXiv.2402.13440

Sujatha, R. (2025). Agentic artificial intelligence: Foundations and architecture. International Journal of Advanced Multidisciplinary Applications. https://www.ijama.in/current-edition/

Sumers, T. R., et al. (2023). Cognitive architectures for language agents. arXiv. doi:10.48550/arXiv.2309.02427

Swartout, W. R. (1985). Rule-based expert systems and the explanation problem. Artificial Intelligence, 26. doi:10.1016/0004-3702(85)90067-0

Tabassi, E. (2023). Artificial intelligence risk management framework (AI RMF 1.0) (NIST AI 100-1). National Institute of Standards and Technology. doi:10.6028/NIST.AI.100-1

Tan, H., Zhang, Z., Ma, C., Chen, X., Dai, Q., & Dong, Z. (2025). MemBench: Towards more comprehensive evaluation on the memory of LLM-based agents. In Findings of the

Association for Computational Linguistics: ACL 2025 (pp. 19336–19352). Association for Computational Linguistics. doi:10.18653/v1/2025.findings-acl.989

Tran, K., et al. (2025). Multi-agent collaboration mechanisms: A survey of LLM-based agents. arXiv. doi:10.48550/arXiv.2501.06322

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., et al. (2017). Attention is all you need. arXiv. doi:10.48550/arXiv.1706.03762

Wang, G., et al. (2023). Voyager: An open-ended embodied agent with large language models. arXiv. doi:10.48550/arXiv.2305.16291

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

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., et al. (2022). Chain-of-thought prompting elicits reasoning in large language models. arXiv. doi:10.48550/arXiv.2201.11903

Westhaußer, J., et al. (2025). CAIM: A cognitive AI memory architecture for long-term interaction. In Proceedings of ACM IUI 2025. doi:10.1145/3742413.3789222

White, J. (2024). Generative and agentic software systems. arXiv. doi:10.48550/arXiv.2408.01768

Wissuchek, C., & Zschech, P. (2025). Exploring agentic artificial intelligence systems: Towards a typological framework. arXiv. doi:10.48550/arXiv.2508.00844

Wooldridge, M., & Jennings, N. R. (1995). Intelligent agents: Theory and practice. Knowledge Engineering Review, 10(2), 115-152. doi:10.1017/S0269888900008122

Wu, D., Wang, H., Yu, W., Zhang, Y., Chang, K.-W., & Yu, D. (2025). LongMemEval: Benchmarking chat assistants on long-term interactive memory. In Proceedings of the International Conference on Learning Representations (ICLR). https://arxiv.org/abs/2410.10813

Wu, Q., Bansal, G., Zhang, J., Wu, Y., Li, B., Zhu, E., ... Wang, C. (2024). AutoGen: Enabling next-gen LLM applications via multi-agent conversation. In Proceedings of the Conference on Language Modeling (COLM). https://www.microsoft.com/en-us/research/publication/autogen-enabling-next-gen-llm-applications-via-multi-agent-conversation-framework/

Wu, S., et al. (2022). AI Chains: Transparent and controllable human-AI interaction by chaining large language model prompts. In Proceedings of CHI 2022. doi:10.1145/3491102.3517582

Wu, T., et al. (2024). StreamBench: Towards benchmarking continuous improvement of language agents. arXiv. doi:10.48550/arXiv.2406.08747

Xi, Z., Chen, W., Guo, X., He, W., Ding, Y., Hong, B., et al. (2025). The rise and potential of large language model based agents: A survey. Science China Information Sciences. doi:10.1007/s11432-024-4222-0

Xie, T., Zhang, D., Chen, J., Li, X., Zhao, S., Cao, R., ... Yu, T. (2024). OSWorld: Benchmarking multimodal agents for open-ended tasks in real computer environments. In Advances in Neural Information Processing Systems, 37. https://proceedings.neurips.cc/paper_files/paper/2024/file/5d413e48f84dc61244b6be550f1cd8f5-Paper-Datasets_and_Benchmarks_Track.pdf

Xu, B. (2026). AI agent systems: Architectures, applications, and evaluation. arXiv. doi:10.48550/arXiv.2601.01743

Yao, S., Shinn, N., Razavi, P., & Narasimhan, K. (2024). τ-bench: A benchmark for tool-agent-user interaction in real-world domains. arXiv preprint arXiv:2406.12045. doi:10.48550/arXiv.2406.12045

Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T. L., Cao, Y., & Narasimhan, K. (2023b). Tree of thoughts: Deliberate problem solving with large language models. arXiv. doi:10.48550/arXiv.2305.10601

Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., & Cao, Y. (2023a). ReAct: Synergizing reasoning and acting in language models. arXiv. doi:10.48550/arXiv.2210.03629

Yehudai, A., et al. (2025). Evaluating AI agents: A survey. arXiv. doi:10.48550/arXiv.2503.16416

Zhang, G., Fu, M., Wan, G., Yu, M., Wang, K., & Yan, S. (2025). G-Memory: Tracing hierarchical memory for multi-agent systems. arXiv. doi:10.48550/arXiv.2506.07398

Zheng, J., et al. (2024). NaturalPlan: Benchmarking LLMs on natural language planning. arXiv. doi:10.48550/arXiv.2406.04520

Zheng, Y., et al. (2026). Lifelong learning for LLM-based agents: A roadmap. IEEE Transactions on Pattern Analysis and Machine Intelligence. doi:10.1109/TPAMI.2025.3650546

Zhou, A., Yan, K., Shlapentokh-Rothman, M., Wang, H., & Wang, Y.-X. (2024). Language agent tree search unifies reasoning, acting, and planning in language models. arXiv. doi:10.48550/arXiv.2310.04406

Zhou, C., Tang, Y., Chen, K., Bai, X., Qi, S., Shen, L., & Zhang, M. (2026). From fragmentation to systematic design: Architecting LLM-based multi-agent systems. Authorea Preprints. doi:10.36227/techrxiv.176827304.41872996/v2

Zhou, S., Xu, F. F., Zhu, H., Zhou, X., Lo, R., Sridhar, A., ... Neubig, G. (2024). WebArena: A realistic web environment for building autonomous agents. In Proceedings of the International Conference on Learning Representations (ICLR). https://proceedings.iclr.cc/paper_files/paper/2024/hash/4410c0711e9154a7a2d26f9b3816d1ef-Abstract-Conference.html

Zhu, K., Du, H., Hong, Z., Yang, X., Guo, S., Wang, Z., ... You, J. (2025). MultiAgentBench: Evaluating the collaboration and competition of LLM agents. In Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Vol. 1, pp. 8580–8622). Association for Computational Linguistics. doi:10.18653/v1/2025.acl-long.421