A Survey on AI Agent Harness

Zhongwei Xie, Xiyu Ren, Tianshi Zheng, Jiaxin Bai, Wei Fan,
Baixuan Xu, Haoran Li, Huihao Jing, Yangqiu Song
Department of Computer Science and Engineering, HKUST, Hong Kong SAR, China
{zxiebk, tzhengad}@connect.ust.hk, yqsong@cse.ust.hk
https://github.com/HKUST-KnowComp/Awesome-Agent-Harness

Abstract

As LLM-based agents advance from simple tool-calling workflows to long-horizon autonomous systems, the infrastructure beyond the base model has become an increasingly decisive factor in agent performance. This infrastructure, broadly termed the **Agent Harness**, has been recognized across recent technical reports and industrial practice as a first-class concern, yet it remains a highly composite and multi-layered concept that resists simple characterization. Understanding the distinct layers and evolutionary dynamics of the Agent Harness is essential for advancing reliable agent systems, but a systematic survey dedicated to this infrastructure is notably absent. Existing surveys predominantly address model-intrinsic reasoning or multi-agent coordination, treating the harness as a trivial implementation detail. To bridge this gap, we propose a **Unified Architectural Taxonomy** that delineates the Agent Harness as a four-layered stack: (1) Execution & Orchestration, (2) Context & Trajectory Management, (3) Interaction Surface & Execution Environment, and (4) Constraints & Guardrails. Our analysis reveals two key findings: the Agent Harness is a widely overlooked performance bottleneck, and it exists in a *co-evolutionary* relationship with the underlying models. We argue that harness architectures must be "Built to Delete" to keep pace with this co-evolutionary cycle.

1 Introduction

The Artificial Intelligence (AI) landscape is transitioning from static question-answering toward dynamic, long-horizon autonomous agents. As agentic applications grow in complexity, the performance bottleneck of AI systems has shifted: it is no longer solely constrained by the "raw intelligence" of underlying Large Language Models (LLMs), but increasingly by the surrounding infrastructure's capacity to maintain reliability over extended execution trajectories. In early iterations, a simple tool-calling *workflow* required minimal infrastructure. However, when an agent transitions to open-ended, multi-day software engineering tasks, it demands orchestration, persistent memory, and fault-tolerant execution. This evolution culminates in complex, highly privileged platforms like OpenClaw (OpenClaw Contributors, 2026), where agents are granted deep system access and the absence of robust infrastructure becomes a critical vulnerability, exposing the system to catastrophic failures and agent injection attacks (Manik and Wang, 2026).

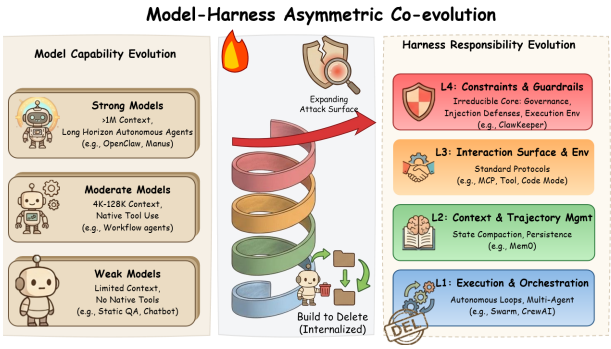

Figure 1: The Asymmetric Co-evolution of Model and Harness.

This infrastructure layer has come to be known as the **Agent Harness**. Recent work has begun to articulate the concept from multiple angles: effective harnesses for long-running agents (Young, 2025), harness engineering as a distinct discipline (He et al., 2026), the harness as a first-class abstraction (Schmid, 2026), end-to-end automated optimization of harness code (Lee et al., 2026), and the emergence of production-grade managed agent platforms (NousResearch, 2026; Anthropic, 2026a). Drawing on these efforts, we systematize the Agent Harness as the *integrated execution layer* that persists across model invocations, manages state beyond any single context window, and enforces behavioral constraints that no model can impose on itself. By this definition, the scope of this survey is bounded by a single criterion: any

infrastructure component that cannot be internalized into model weights, and that must exist as an independent runtime layer, belongs to the Agent Harness. This criterion naturally excludes transient prompting strategies and fragmented tool-calling libraries, while focusing on systems that provide a deterministic substrate for the stochastic nature of foundation models.

The Agent Harness, however, is a highly composite concept that spans orchestration, context management, execution environments, and safety guardrails. Understanding its distinct layers and their evolutionary dynamics is essential for advancing reliable agent systems, yet a systematic survey dedicated to this infrastructure is notably absent. Existing surveys predominantly focus on model-intrinsic reasoning strategies or abstract multi-agent interactions (Ferrag et al., 2025), treating the underlying execution infrastructure as a trivial implementation detail.

To bridge this gap, we propose a **Unified Architectural Taxonomy** that delineates the Agent Harness as a four-layered stack: (1) Execution & Orchestration, (2) Context & Trajectory Management, (3) Interaction Surface & Execution Environment, and (4) Constraints & Guardrails. This framework categorizes systems not by their underlying LLMs, but by their structural progression—from the core autonomous execution loop outward to systemic constraints and environmental boundaries. Grounded in this taxonomy, we survey over 150 recent works spanning both academic literature and production systems to map the current landscape. Our survey methodology deliberately incorporates influential industrial artifacts, as many critical harness innovations originate in production systems before academic formalization.

Our analysis yields two key findings. First, the Agent Harness is a widely overlooked performance bottleneck: swapping the same model into a different harness can yield dramatically different results, yet current benchmarks evaluate the monolithic Model-Harness system, falsely attributing infrastructural successes or failures entirely to the model's weights. This conflation has catalyzed a severe *evaluation crisis* that systematically misdirects research investment toward model scaling while the infrastructure bottleneck remains underexplored. Second, Model and Harness exist in a **co-evolutionary** relationship (Figure 1): as models grow more capable (e.g., longer context windows, native tool calling), they absorb simpler harness functions, rendering them obsolete; yet simultaneously, each new frontier of agentic application demands infrastructure capabilities that no model can internalize, such as durable execution, deterministic recovery, and cryptographic permission scoping. The sole exception is the safety and guardrails layer, whose importance grows monotonically with model capability. More capable models unlock higher-autonomy applications that inherently expand the attack surface. Meanwhile, the very properties that necessitate external guardrails, such as output non-determinism, susceptibility to adversarial prompts, and the absence of verifiable intent, are intrinsic to the generative paradigm and are less amenable to self-policing by the very model whose outputs require oversight.

Building on these findings, this survey makes three contributions: (1) a systematic definition and scope delineation of the Agent Harness; (2) a Unified Architectural Taxonomy organizing the harness into a four-layered stack (Section 2), with each layer surveyed in detail (Sections 3–6); and (3) a critical analysis of the evaluation crisis and a forward-looking roadmap arguing that harness architectures must be "Built to Delete" to keep pace with the co-evolutionary cycle (Section 7).

2 Taxonomy of Agent Harness: A Unified Architectural Stack

Figure 2 illustrates the architectural abstraction layering in agent harness frameworks with their associated features. In this section, we discuss their applications and characteristics in more detail.

Layer 1: Execution and Orchestration. Acting as the temporal engine of the harness, Layer 1 is strictly responsible for driving the execution graph and managing the locus of control, operating independently of the semantic content of underlying data payloads and without defining security policies itself. This layer encapsulates the core autonomous event loop that drives iterative reasoning, governed by dynamic routing mechanisms that intelligently dispatch tasks to optimal LLM backends based on cost and capability heuristics. Moving beyond single-actor execution, it treats agents as composable entities, orchestrating the concurrent spawning of specialized subagents and synchronizing state handoffs based on task topologies. Additionally, Layer 1 implements targeted resilience mechanisms to prevent execution halts during reasoning dead-ends or API failures. These compo-

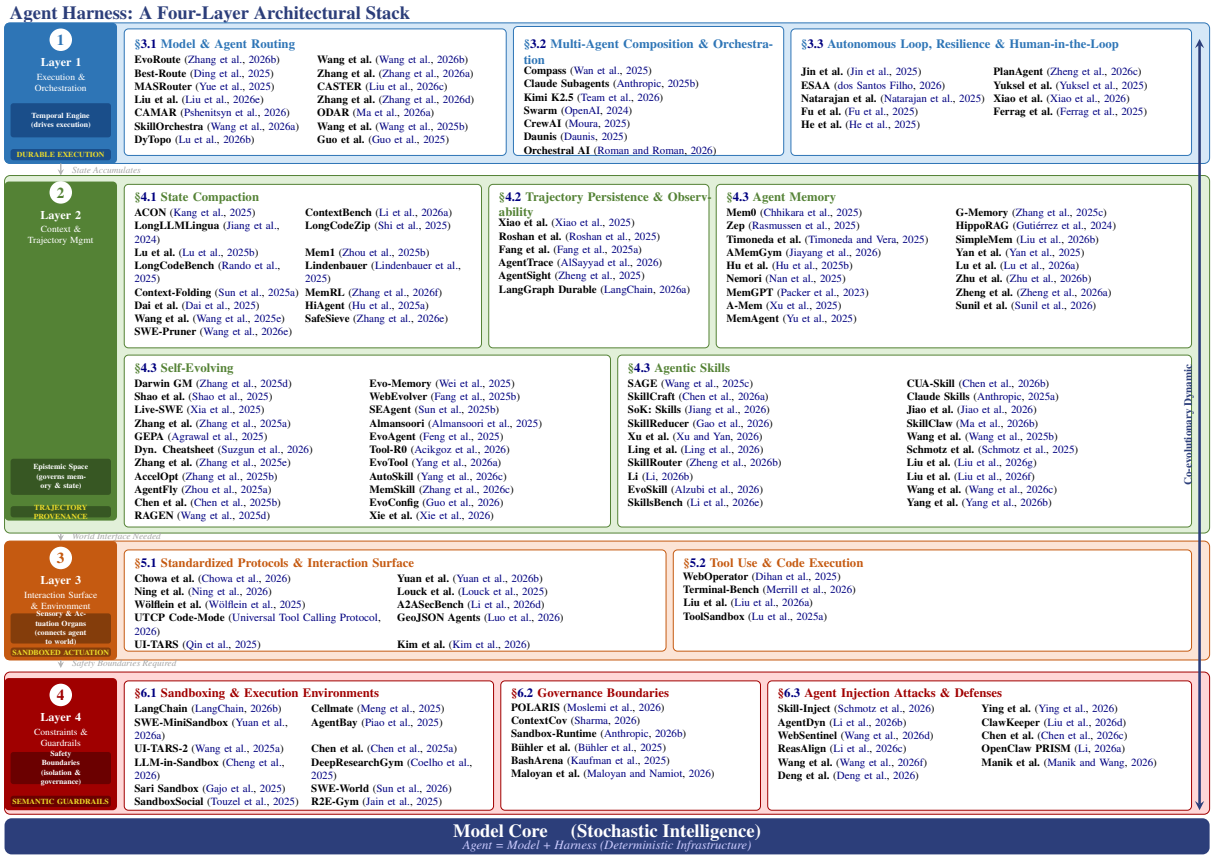

Figure 2: Agent Harness: A Four-Layer Architectural Stack. Each pill lists representative works in **bold** with citation keys; all citations are hyperlinked to the bibliography.

nents include deterministic retry logic, exponential backoffs, and error recovery routing, which collectively ensure continuous operation. Crucially, this layer facilitates a continuous evolutionary cycle by coupling functionally disparate layers: it processes discrete feedback signals from security modules to reroute control flow and triggers semantic garbage collection to prevent entropy accumulation.

Layer 2: Context and Trajectory Management. While the orchestration layer manages execution time, Layer 2 governs the agent's *epistemic space*. Its primary function is to mitigate the fundamental bottlenecks of context window saturation and catastrophic forgetting, all while maintaining strict system observability. To prevent performance degradation and the onset of "context rot," this layer employs sophisticated prompt layout engines and semantic-aware eviction strategies. It sustains lifecycle continuity through self-evolving memory architectures, managing cross-session checkpointing and state serialization. Layer 2 secures the agent's working memory and execution pathways by persisting them into external databases or vector stores. Simultaneously, it securely decouples telemetry from the model's active memory, logging tool invocations and state mutations into an immutable trace graph for compliance and post-hoc debugging.

Layer 3: Interaction Surface and Execution Environment. Because language models are inherently disembodied, Layer 3 constitutes the *sensory and actuation organs* of the agentic system. This layer defines the environmental footprint of the agent, standardizing the interfaces for discrete functional tool calling and arbitrary code execution, effectively translating abstract model intent into concrete environmental state changes. To manage these interactions dynamically without saturating the initialization context, it relies on delayed-discovery protocols, such as the Model Context Protocol (MCP), employing progressive disclosure.

Layer 4: Constraints and Guardrails. Finally, because LLM outputs are inherently probabilistic, Layer 4 enforces deterministic **safety boundaries** on the system at two levels: *physical isolation* through hardware or kernel-level sandboxing to contain erratic or adversarial agent behavior, and *logical governance* through pre- and post-

execution validation, including schema-enforced format checks, static code analysis, and runtime semantic monitoring. This layer serves as the primary defense against adversarial prompt and agent injections, enforcing strict principle-of-least-privilege access controls on all tool invocations. By running parallel input-output filtering, Layer 4 ensures that policy deviations trigger immediate "fail-fast" rollbacks, preventing catastrophic side effects.

By utilizing this unified, layered taxonomy, researchers and infrastructure engineers can precisely map the operational boundaries of existing frameworks and identify where a specific system's bottlenecks lie, facilitating the targeted evaluation metrics discussed in subsequent sections.

3 Layer 1: Execution & Orchestration Layer

3.1 Model & Agent Routing

The foundational responsibility of Layer 1 is to drive the agent's **autonomous execution loop**, which is the continuous cycle of observing the environment, reasoning about the next step, and executing tool calls. A critical dimension of this loop is *model routing*: dynamically determining which LLM or specialized agent should handle a given subtask. As model ecosystems diversify, static assignment becomes suboptimal. EvoRoute (Zhang et al., 2026b) introduces experience-driven self-routing, while Best-Route (Ding et al., 2025) optimizes for test-time compute efficiency. MAS-Router (Yue et al., 2025) learns to route LLMs specifically for multi-agent systems. Routing strategies have expanded to encompass multiple dimensions: SkillOrchestra (Wang et al., 2026a) routes agents via skill transfer, DyTopo (Lu et al., 2026b) employs dynamic topology routing via semantic matching, and CASTER (Liu et al., 2026c) breaks the cost-performance barrier in multi-agent orchestration. Budget-aware routing (Zhang et al., 2026a) and ODAR's active inference routing (Ma et al., 2026a) focus on resource constraints. Adaptive VLM routing for computer-use agents is addressed in (Liu et al., 2026e), CAMAR (Pshenitsyn et al., 2026) handles continuous-action multi-agent routing, and query-aware budget-tier routing for runtime agent memory is proposed in (Zhang et al., 2026d). Additional frameworks include optimal agent selection (Wang et al., 2025b) and generalized routing for model-agent orchestration (Guo et al., 2025). For multi-agent systems operating under resource constraints, confidence-aware routing (Wang et al., 2026b) dynamically allocates tasks across models of varying scale based on estimated task difficulty, reducing cost without sacrificing quality.

3.2 Multi-Agent Composition & Orchestration

Modern agent harnesses increasingly treat agents as **composable, modular entities** rather than monolithic systems. This subsection examines how Layer 1 orchestrates concurrent subagent spawning, delegation, and synchronized state handoffs. Two dominant composition patterns have emerged. The first is *lightweight, educational orchestration*, exemplified by OpenAI's Swarm (OpenAI, 2024), which models handoffs as explicit function-call transfers between peer agents with minimal overhead. The second is *hierarchical subagent architectures*, as seen in Anthropic's subagent system (Anthropic, 2025b) and Kimi K2.5's visual agentic intelligence (Team et al., 2026), where a parent agent dynamically spawns specialized child agents and aggregates their results. Compass (Wan et al., 2025) maintains an evolving context representation that prevents the loop from losing coherence over long horizons in multi-agent settings.

Beyond these architectural patterns, orchestrating multiple agents requires robust framework-level support to manage communication and task delegation. At the framework level, CrewAI (Moura, 2025) provides role-based orchestration for teams of autonomous agents, while Orchestral AI (Roman and Roman, 2026) proposes a general-purpose orchestration framework with formalized communication protocols. A declarative language for specifying agent workflows, separating orchestration logic from agent implementation, is introduced in (Daunis, 2025).

3.3 Autonomous Loop, Resilience & Human-in-the-Loop

Beyond routing and composition, the autonomous loop itself must be resilient to non-termination and drift. When the system encounters reasoning dead-ends, API failures, or environmental exceptions, Layer 1 executes deterministic resilience mechanisms to ensure continuous forward momentum. These include structured retry logic with exponential backoffs, error-classification-based recovery routing, and graceful degradation to lower-capability models when primary endpoints are un-

available (Jin et al., 2025). Event-sourced architectures (dos Santos Filho, 2026) provide an additional recovery primitive: because every state transition is logged as an immutable event, the system can deterministically replay execution from any prior checkpoint, enabling precise failure diagnosis and branch-level recovery. The degree of human involvement in the loop spans a wide spectrum: from full human-in-the-loop oversight (Natarajan et al., 2025; Fu et al., 2025) to test-time self-improvement with only intermittent guidance (He et al., 2025), and to fully closed-loop planning agents such as PlanAgent (Zheng et al., 2026c), reflecting the autonomy tiers discussed in our taxonomy. Critically, these recovery mechanisms operate as a **cross-layer feedback loop** (Yuksel et al., 2025) which enables self-correcting workflows (Xiao et al., 2026) where the harness autonomously detects, diagnoses, and recovers from failures without human intervention. As models internalize stronger self-correction capabilities, simple orchestration loops become redundant, yet multi-agent compositions and cross-model routing create *new*, inherently infrastructural orchestration demands. This shifts Layer 1's center of gravity from compensating for model weaknesses to enabling system-level capabilities that transcend any individual model.

4 Layer 2: Context & Trajectory Management Layer

4.1 State Compaction

As agent trajectories grow, the fixed-size context window becomes a critical bottleneck, a phenomenon we term **Context Rot**, where accumulated irrelevant tokens progressively degrade reasoning quality. Layer 2 addresses this through sophisticated **state compaction** strategies. A foundational line of work focuses on *prompt compression*: LongLLMLingua (Jiang et al., 2024) accelerates long-context processing via token-level importance scoring and selective eviction, while LongCodeZip (Shi et al., 2025) applies domain-specific compression for code contexts. ACon (Kang et al., 2025) optimizes compression specifically for long-horizon agent tasks, balancing information retention against token budget.

A complementary approach is *agent-aware context pruning*, which leverages the structure of agentic execution to identify dispensable content. SWE-Pruner (Wang et al., 2026e) performs self-adaptive pruning for coding agents by learning which repository context is task-relevant, and SafeSieve (Zhang et al., 2026e) applies progressive pruning to multi-agent communication channels. Notably, the "Complexity Trap" is demonstrated in (Lindenbauer et al., 2025): simple observation masking can match the effectiveness of expensive LLM-based summarization for agent context management, challenging the assumption that more sophisticated compression is always better.

For longer horizons, *context folding and scaling* methods restructure rather than discard information. Context-folding (Sun et al., 2025a) hierarchically summarizes historical trajectories, while summarization-based context management (Lu et al., 2025b) integrates end-to-end RL to learn what to retain. Mem1 (Zhou et al., 2025b) synergizes memory with reasoning to maintain efficiency, and pretraining context compressors with embedding-based memory is proposed in (Dai et al., 2025). HiAgent (Hu et al., 2025a) introduces hierarchical working memory management for solving long-horizon agent tasks. The evaluation of these strategies is supported by dedicated benchmarks: ContextBench (Li et al., 2026a) and SWE Context Bench (Zhu et al., 2026a) measure context retrieval quality in coding agents, and LongCodeBench (Rando et al., 2025) evaluates performance at the 1M-token scale.

4.2 Trajectory Persistence & Observability

Beyond in-context management, Layer 2 must **persist** the agent's execution history to external storage for recovery, replay, and continuous learning. This involves two core primitives. *Trajectory reduction* (Xiao et al., 2025) compresses execution traces into compact representations suitable for storage and retrieval, while *semantic checkpointing* (Roshan et al., 2025) captures not just code state but the agent's epistemic context, including reasoning chains and environmental snapshots, at designated points. Notebook checkpointing with AI agents at scale is evaluated in (Fang et al., 2025a), validating the practical benefits of trajectory persistence for reproducibility. At the infrastructure level, *durable execution* runtimes such as LangGraph's durable execution layer (LangChain, 2026a) provide fault-tolerant, long-running execution with built-in checkpointing, enabling agents to survive process restarts and resume from their last consistent state. This capability is essential for the multi-hour and multi-day agent sessions increasingly common in software engineering and

research tasks.

Layer 2 also decouples **observability** from the model's working memory. Rather than burdening the context window with execution logs, the harness securely records every tool invocation, state mutation, and API call into an immutable, structured trace graph. AgentTrace (AlSayyad et al., 2026) provides a structured logging framework specifically designed for agent system observability, while AgentSight (Zheng et al., 2025) leverages eBPF for system-level, low-overhead monitoring of agent processes. These telemetry systems ensure the "black box" remains fully transparent to human operators for compliance, post-hoc debugging, and performance analysis, without polluting the agent's epistemic state. It is worth noting that Layer 2 is perhaps the most sensitive to Model-Harness co-evolution: larger context windows (from 4K to 1M+ tokens) have absorbed some compression needs, yet paradoxically *increase* the demand for trajectory management, longer contexts enable longer agent sessions, which generate more state to persist, audit, and compact. As models gain native memory capabilities, the harness role shifts from *providing* memory to *governing* it: ensuring provenance, preventing poisoning, and managing cross-session consistency.

4.3 Agent Memory and Self-evolving Architecture

Memory Systems. Agent memory extends trajectory persistence into a structured, queryable knowledge layer. The landscape ranges from production-ready platforms to research prototypes. Mem0 (Chhikara et al., 2025) provides scalable long-term memory with automatic extraction and retrieval, while Zep (Rasmussen et al., 2025) organizes agent memory as a temporal knowledge graph. At the research frontier, MemGPT (Packer et al., 2023) pioneers an OS-inspired memory hierarchy that pages information between fast (context window) and slow (external storage) tiers. A-Mem (Xu et al., 2025) introduces agentic memory that self-organizes based on usage patterns, HippoRAG (Gutiérrez et al., 2024) draws on neuroscience (specifically hippocampal indexing theory) for neurobiologically inspired long-term retrieval, and Nemori (Nan et al., 2025) proposes self-organizing memory inspired by cognitive science. SimpleMem (Liu et al., 2026b) demonstrates that long horizon memory can be achieved with surprisingly simple architectures, and general agentic memory via deep research is explored in (Yan et al., 2025).

For multi-agent settings, G-Memory (Zhang et al., 2025c) traces hierarchical memory structures across agent teams, and MemAgent (Yu et al., 2025) reshapes long-context processing via multi-conversation RL-based memory agents. The question of *how* to remember is itself an active research area: Adaptive memory structures that dynamically select storage strategies are studied in (Lu et al., 2026a), while provenance-aware tiered memory that trades lossy compression for verified recall is proposed in (Zhu et al., 2026b). The effect of model memory on LLM performance in annotation tasks is demonstrated in (Timoneda and Vera, 2025), and memory evaluation through incremental multi-turn interactions is conducted in (Hu et al., 2025b). A roadmap for lifelong learning of LLM-based agents is provided in (Zheng et al., 2026a). Evaluation is supported by interactive benchmarks such as AMemGym (Jiayang et al., 2026), which tests memory in long-horizon conversational settings. The security of these memory systems is also a growing concern, as memory poisoning attacks against memory-augmented agents are demonstrated in (Sunil et al., 2026).

Self-Evolving Architectures. A defining aspiration of Layer 2 is enabling agents that **improve over their operational lifetime** without retraining the underlying model. The Darwin Godel Machine (Zhang et al., 2025d) explores open-ended self-modification, while Live-SWE-agent (Xia et al., 2025) demonstrates on-the-fly self-evolution for software engineering tasks. EvoAgent (Feng et al., 2025) maintains a continual world model that adapts as the agent encounters new domains, and agentic context engineering (Zhang et al., 2025e) proposes evolving the context construction strategy itself. RL-based approaches, including MemRL (Zhang et al., 2026f) and RAGEN (Wang et al., 2025d), use episodic reinforcement learning to refine memory and retrieval policies through experience.

The self-evolution paradigm has been explored across multiple dimensions. *Prompt and strategy evolution* includes GEPA (Agrawal et al., 2025), which shows reflective prompt evolution can outperform reinforcement learning, and dynamic cheatsheets (Suzgun et al., 2026) for test-time learning with adaptive memory. *Domain-specific self-improvement* has been demonstrated

for ML library development (Zhang et al., 2025a), AI accelerator kernel optimization (Zhang et al., 2025b), and clinical interactions (Almansoori et al., 2025). AgentFly (Zhou et al., 2025a) achieves agent fine-tuning without fine-tuning the underlying LLM. *Co-evolutionary approaches* include multi-agent evolution through co-evolution (Chen et al., 2025b), WebEvolver's co-evolving world model (Fang et al., 2025b), SEAgent (Sun et al., 2025b), and EvoConfig's self-evolving multi-agent configuration (Guo et al., 2026). *Tool-oriented self-evolution* is addressed by Tool-R0 (Acikgoz et al., 2026), which learns tool use from zero data, and EvoTool (Yang et al., 2026a), which optimizes tool-use policies via blame-aware mutation. MemSkill (Zhang et al., 2026c) learns and evolves memory skills, and Evo-Memory (Wei et al., 2025) benchmarks test-time self-evolving memory. The failure mode of over-searching in search-augmented LLMs is analyzed in (Xie et al., 2026). However, self-evolution introduces risks: emergent misevolution, where agents develop unintended behaviors through unconstrained self-modification, is documented in (Shao et al., 2025).

Agentic Skills. A closely related paradigm is the acquisition and management of **agentic skills**, which are reusable, composable behavioral modules that extend an agent's capabilities without modifying its weights. A systematization of knowledge on agentic skills, distinguishing them from simple tool use, is provided in (Jiang et al., 2026), and the architecture, acquisition, and security of agent skills are surveyed in (Xu and Yan, 2026). Major platforms have embraced this paradigm: Anthropic's Agent Skills launch (Anthropic, 2025a) integrates skill systems into production agents, while SkillClaw (Ma et al., 2026b) enables collective skill evolution across agent clusters through cloud-based sharing. A data-driven analysis of Claude's skill ecosystem is provided in (Ling et al., 2026).

Skill acquisition and routing has been studied extensively. SkillCraft (Chen et al., 2026a) studies whether agents can learn to use tools skillfully through practice, SkillRouter (Zheng et al., 2026b) addresses efficient skill selection at scale via retrieve-and-rerank, and SkillReducer (Gao et al., 2026) optimizes skills for token efficiency. AutoSkill (Yang et al., 2026c) enables experience-driven lifelong learning through skill self-evolution, EvoSkill (Alzubi et al., 2026) automates skill discovery for multi-agent systems, and programmatic skills for agentic tasks are induced in (Wang et al., 2025e). CUA-Skill (Chen et al., 2026b) develops skills for computer-using agents, reasoning is enhanced via compositional skill synthesis in (Jiao et al., 2026), and SAGE (Wang et al., 2025c) provides a broader framework for skill-augmented generation. When single agents with skills can replace multi-agent systems and when they fail is investigated in (Li, 2026b), while SkillsBench (Li et al., 2026e) provides a dedicated benchmark for evaluating skill effectiveness across diverse tasks.

5 Layer 3: Interaction Surface & Execution Environment

5.1 Standardized Protocols & Interaction Surface

Because LLMs are inherently disembodied, Layer 3 provides the *sensory and actuation organs* through which agents interact with the external world. A critical design challenge is **protocol standardization**: how should the harness expose tools, APIs, and environments to the model? The dominant approach relies on delayed-discovery protocols such as the Model Context Protocol (MCP), which binds external APIs dynamically and employs progressive disclosure to prevent context saturation during initialization. However, the security implications of such open protocols are non-trivial: it has been demonstrated that the MCP specification itself introduces prompt injection vulnerabilities in tool-integrated agents (Maloyan and Namiot, 2026). Alternative approaches include UTCP's Code-Mode (Universal Tool Calling Protocol, 2026), which provides a plug-in architecture for extending agent capabilities through structured code interfaces. Beyond tool-binding protocols, a parallel standardization effort targets **agent-to-agent (A2A) communication**. Current multi-agent message passing remains largely ad hoc; Yuan et al. (2026b) argue that moving beyond raw message passing toward semantically aligned agent communication is essential for robust multi-agent coordination. Google's A2A protocol represents a concrete step in this direction, though its security properties require careful scrutiny: Louck et al. (2025) identify vulnerabilities in the A2A specification related to sensitive data exposure and propose protocol-level mitigations, while A2ASecBench (Li et al., 2026d) provides the first protocol-aware security benchmark for evaluating agent-to-agent multi-agent systems.

5.2 Tool Use & Code Execution

The **tool-use interface** defines how agents discover, select, and invoke functional tools. A comprehensive review of the evolution from language understanding to action execution in LLM agents is provided in (Chowa et al., 2026). A notable trend is *agent-driven tool creation*: agents can autonomously synthesize new tools when existing ones are insufficient (Wölflein et al., 2025). Tool reliability is equally critical; methods for detecting defects in LLM-based agent tools before deployment are proposed in (Ning et al., 2026). For agents operating under resource constraints, intention-based planning that optimizes tool invocation sequences under explicit budget limits is introduced in (Liu et al., 2026a). ToolSandbox (Lu et al., 2025a) provides a stateful, conversational benchmark for evaluating tool-use capabilities in realistic multi-turn settings, and agent robustness when APIs deviate from their idealized specifications is evaluated in (Kim et al., 2026).

While functional APIs provide structured capabilities, many complex tasks demand a more open-ended interaction modality. Beyond discrete tool calls, many agentic tasks require **arbitrary code execution** in terminal or REPL environments. Terminal-Bench (Merrill et al., 2026) benchmarks agents on hard, realistic command-line tasks, revealing that current agents struggle with stateful terminal interactions that require maintaining environmental context across commands. For GUI-based interaction, UI-TARS (Qin et al., 2025) pioneers native GUI agents that interact directly with visual interfaces, while WebOperator (Dihan et al., 2025) employs action-aware tree search for autonomous web navigation. Two paradigms for geospatial agents, function calling versus code generation, are compared in (Luo et al., 2026), finding that the optimal interaction modality depends on task complexity. These diverse interaction surfaces underscore the need for Layer 3 to provide a *unified abstraction* over heterogeneous execution environments. Indeed, as models gain native function-calling and code-generation capabilities, simple tool interfaces become redundant, yet complex, OpenClaw-class applications demand richer sandboxing, protocol standardization, and multi-modal interaction surfaces that only the harness can provide. The evolution from ReAct-style single tool calls to full terminal and GUI execution illustrates this clearly: each advance in model capability unlocks more ambitious applications, which in turn require more sophisticated execution environments.

6 Layer 4: Constraints & Guardrails Layer

6.1 Sandboxing & Execution Environments

Layer 4 enforces strict **hardware or kernel-level isolation** to contain erratic or adversarial agent behavior. The design space for agent sandboxing has been formalized in (LangChain, 2026b), identifying two fundamental patterns: *agent-in-sandbox* (the agent runs inside a constrained container) and *sandbox-as-tool* (the agent invokes sandboxed environments as external resources). Anthropic's sandbox-runtime (Anthropic, 2026b) exemplifies production-grade isolation for agent execution, while SWE-MiniSandbox (Yuan et al., 2026a) and SWE-World (Sun et al., 2026) explore container-free and Docker-free alternatives that reduce overhead. LLM-in-sandbox architectures are shown to elicit general agentic intelligence through environmental grounding (Cheng et al., 2026).

The sandbox paradigm extends across diverse application domains. For web agents, Cellmate (Meng et al., 2025) sandboxes browser AI agents, and DeepResearchGym (Coelho et al., 2025) provides a reproducible evaluation sandbox for deep research. AgentBay (Piao et al., 2025) provides a hybrid interaction sandbox enabling seamless human-AI intervention. Domain-specific sandboxes include Sari Sandbox (Gajo et al., 2025) for virtual retail environments and SandboxSocial (Touzel et al., 2025) for simulating social media dynamics with multimodal agents. R2E-Gym (Jain et al., 2025) provides procedural environments with hybrid verifiers for scaling open-weight SWE agents. Importantly, static sandboxes are argued to be fundamentally inadequate for modeling societal complexity, with open-ended co-evolutionary environments advocated as an alternative (Chen et al., 2025a). UI-TARS-2 (Wang et al., 2025a) advances GUI agent capabilities through multi-turn reinforcement learning within sandboxed environments.

6.2 Governance Boundaries

Because LLM outputs are inherently probabilistic, Layer 4 imposes deterministic **governance boundaries** on every action before and after execution. Pre-execution constraints include schema-enforced constrained decoding, static code analysis, and plan validation. POLARIS (Moslemi et al., 2026)

demonstrates typed planning with governed execution for back-office automation, where each agent action is validated against a formal type system before being dispatched. ContextCov (Sharma, 2026) takes a complementary approach: it derives executable constraints directly from natural-language instruction files, automatically enforcing coverage requirements that would otherwise depend on manual review.

While pre-execution constraints ensure structural validity, runtime governance must restrict what the agent is allowed to access. Effective permission management for agents requires moving beyond static access control models designed for human users. Anthropic's sandbox-runtime (Anthropic, 2026b) enforces execution-time permission boundaries, while a comprehensive framework for securing agent execution across the full lifecycle is proposed in (Bühler et al., 2025). BashArena (Kaufman et al., 2025) provides a controlled setting for studying highly privileged agents, those with root-level system access, revealing the catastrophic potential of insufficient permission scoping.

Traditional access control paradigms, however, struggle to accommodate the dynamic nature of autonomous execution. The core challenge is that agents operate with *transient authority*: their need for a specific resource is contingent on a fleeting sub-task, not a persistent role. Current RBAC models over-provision by design. As discussed in our Future Directions (Section 7), we argue for a transition toward **Intent-Based Access Control (IBAC)**, where permissions are dynamically scoped to the agent's verified reasoning chain.

6.3 Agent Injection Attacks and Defenses

The expanded interaction surface of agentic systems introduces a critical vulnerability class: **agent injection**, where adversarial content in the environment manipulates agent behavior. The security implications of the skill paradigm are particularly significant. Beyond skill injection attacks (Schmotz et al., 2025), a large-scale empirical study of security vulnerabilities in agent skills in the wild is conducted in (Liu et al., 2026f), and hidden-comment injection where skills contain concealed malicious instructions is demonstrated in (Wang et al., 2026c). Most concerning, "zombie agents" achieving persistent control of self-evolving LLM agents via self-reinforcing injections is shown in (Yang et al., 2026b), and the prevalence of malicious agent skills is further documented in (Liu et al., 2026g). This threat operates at multiple levels. At the *skill level*, Skill-Inject (Schmotz et al., 2026) shows that agent skill files, intended as benign behavioral extensions, can serve as a trivially exploitable injection vector. AgentDyn (Li et al., 2026b) provides a dynamic, open-ended benchmark for evaluating the robustness of real-world agent security systems against these attacks.

A concentrated body of work around the OpenClaw platform illustrates the full attack-defense lifecycle. Offensive studies formalize attacks on personalized local agents (Wang et al., 2026f; Manik and Wang, 2026), while defensive responses include ClawKeeper (Liu et al., 2026d), a comprehensive safety layer combining skill validation, plugin sandboxing, and runtime watchers, and OpenClaw PRISM (Li, 2026a), a zero-fork, defense-in-depth runtime security layer. Trajectory-based auditing (Chen et al., 2026c) offers a post-hoc safety mechanism by analyzing execution traces for anomalous patterns. These findings generalize beyond any single platform: they demonstrate the fundamental tension between agent extensibility and security that every harness must navigate.

For web-facing agents, WebSentinel (Wang et al., 2026d) detects and localizes prompt injection attacks in real time, while ReasAlign (Li et al., 2026c) enhances safety alignment specifically against injection attacks through reasoning-enhanced alignment training. At the system level, Layer 4 runs **parallel input/output filtering and security checks** as an independent governance layer; if a policy deviation is detected, it triggers a "fail-fast" rollback before catastrophic side effects occur. The mechanisms required to tame autonomous agent threats are analyzed in (Deng et al., 2026), and a systematic study of security threat architectures and defense patterns is provided in (Ying et al., 2026). These concurrent guardrails complement governance boundaries: while the latter enforce structural validity, runtime monitors detect semantic policy violations, such as attempts at unauthorized data exfiltration or lateral movement, that cannot be caught through static analysis alone. Crucially, safety governance cannot be delegated to the very system being governed. As more capable models unlock higher-autonomy applications, they inherently expand the attack surface, making Layer 4 an irreducible and increasingly vital component of the Agent Harness.

7 Challenges and Future Directions

Across the four layers surveyed, a consistent pattern emerges: the Agent Harness is not a static scaffold but a co-evolutionary partner to the underlying model. Yet this co-evolution is fundamentally asymmetric. For Layers 1 through 3 (Orchestration, Context, and Environment), the relationship follows a "Bitter Lesson" logic: as base models inevitably internalize higher-order planning, native tool calling, and extended context windows, the compensatory mechanisms built into the harness become obsolete. A robust Harness in these layers must be "Built to Delete." Conversely, Layer 4 (Constraints & Guardrails) exhibits a monotonically increasing relationship with model capability. As more capable models unlock higher-autonomy applications, they inherently expand the attack surface, making Layer 4 the irreducible core of the Agent Harness. Two critical challenges and future directions follow directly from this asymmetry.

7.1 The Evaluation Crisis: Disentangling Model from Harness.

Prevailing benchmarks evaluate the monolithic agent system, conflating the inherent capability of the model with the capability of its surrounding Harness. Swapping the same model into a different harness yields dramatically different results, yet harness improvements remain invisible to model-centric leaderboards (Lee et al., 2026). This conflation is compounded by two symptoms: (1) a *reproducibility crisis*, where transient network states, residual inter-run interference, and rigid string-matching protocols produce false negatives and unscientific variance; and (2) *Model-Harness overfitting*, where post-training (RLHF/DPO) on a specific harness's tool schemas causes catastrophic generalizability loss in heterogeneous deployments. Addressing this requires a disentangling model from harness of evaluation, which measures absolute reliability over k continuous execution steps, as well as quantifying capabilities extrinsic to the base model, such as a Harness's ability to compress historical trajectories, shed redundant execution logs, and clear 'AI Slop' (hallucinated states or irrelevant artifacts).

7.2 Toward Next-Generation Agent Infrastructure.

Looking ahead, this asymmetric co-evolution drives a fundamental transition in computing infrastructure. As models absorb capabilities that were once the Harness's domain, the infrastructure must not cling to deprecated responsibilities but instead evolve toward new primitives optimized for autonomous agents. We identify four emerging pillars of this transition:

Beyond Git (Semantic Trajectory Tracking). Traditional version control systems (like Git's diffing algorithms) are fundamentally inadequate for tracking non-deterministic AI agent execution. Because these agents lack inherent memory between execution steps, the future requires Semantic Checkpointing systems. Beyond merely recording code changes, these systems must capture the agent's epistemic context, hidden reasoning trees, and the environmental state at the moment of generation. This allows for precise state reconstruction, enabling developers to debug not just the code, but the non-deterministic "thought process" that led to a specific failure or success.

Durable Execution. We must transcend the stateless assumptions of modern Serverless architectures. As agent execution horizons stretch from minutes to days, the infrastructure's core responsibility shifts from aiding "thought" to ensuring survival. Agents require long-running, fault-tolerant runtimes capable of surviving API rate limits, network partitions, and reasoning dead-ends. This necessitates primitives that allow execution states to be explicitly paused, serialized, replayed, and forked mid-execution to explore alternative planning branches without losing progress.

From Manual Forking to Autonomous Semantic Isolation. Current state-of-the-art infrastructures provide mechanisms for session forking and containerization to prevent state contamination. However, these operations remain primarily human-driven, relying on the operator to manually initiate isolation. We argue that a next-generation Agent Harness must internalize this governance. The infrastructure should possess semantic awareness, the ability to autonomously synthesize ephemeral, task-specific sandboxes by anticipating the potential "blast radius" of an agent's intent. By shifting the responsibility of environmental partitioning from the human operator to the Harness, we enable a robust, self-governing runtime capable of enforcing the principle of least privilege at a granular, sub-task level.

From Static RBAC to Intent-Based Access Control (IBAC). Current security models (e.g., IAM, RBAC) are designed for persistent human identities with predictable access patterns. However, agents operate with "transient authority," where their need to access a resource (e.g., a production database) is contingent upon a specific, fleeting sub-task. We identify a critical gap in Least-Privilege Orchestration: the infrastructure must evolve to support Intent-Based Access Control, where permissions are cryptographically bound to the agent's verified reasoning chain. By analyzing the agent's plan ante-hoc, the Harness should dynamically mint short-lived, narrow-scoped tokens that expire immediately upon task completion, preventing "agentic over-privilege" and lateral movement within the system.

8 Conclusion

This survey has systematized the Agent Harness as a four-layered architectural stack and revealed a fundamental asymmetry in its co-evolution with foundation models. As agents move from prototypes to production, the Harness will determine not only performance, but survivability.

Limitations

This survey has several limitations. First, our scope is deliberately constrained to the *infrastructure* dimension of agentic systems; we do not provide an exhaustive treatment of model-intrinsic capabilities such as reasoning or planning, except insofar as they interact with harness design. Second, the Agent Harness landscape is evolving rapidly, many systems discussed here are open-source projects or preprints whose architectures may change significantly between the time of writing and publication. Third, our four-layer taxonomy, while designed to be comprehensive, necessarily imposes categorical boundaries on systems that often exhibit cross-layer integration; the placement of individual works reflects our best judgment but is not always unambiguous. Finally, our evaluation of commercial and closed-source systems (e.g., proprietary coding agents) is limited to publicly available documentation and technical reports.

References

Emre Can Acikgoz, Cheng Qian, Jonas Hübotter, Heng Ji, Dilek Hakkani-Tür, and Gokhan Tur. 2026. Tool-r0: Self-evolving llm agents for tool-learning from zero data. *arXiv preprint arXiv:2602.21320*.

Lakshya A Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziems, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J Ryan, Meng Jiang, et al. 2025. Gepa: Reflective prompt evolution can outperform reinforcement learning. *arXiv preprint arXiv:2507.19457*.

Mohammad Almansoori, Komal Kumar, and Hisham Cholakkal. 2025. Self-evolving multi-agent simulations for realistic clinical interactions. *arXiv preprint arXiv:2503.22678*.

Adam AlSayyad, Kelvin Yuxiang Huang, and Richik Pal. 2026. Agenttrace: A structured logging framework for agent system observability. In *LLM-based Multi-Agent Systems: Towards Responsible, Reliable, and Scalable Agentic Systems*.

Salaheddin Alzubi, Noah Provenzano, Jaydon Bingham, Weiyuan Chen, and Tu Vu. 2026. Evoskill: Automated skill discovery for multi-agent systems. *arXiv preprint arXiv:2603.02766*.

Anthropic. 2025a. Introducing agent skills. https://www.anthropic.com/news/skills. Product Announcement, Oct 2025.

Anthropic. 2025b. Subagents. https://docs.anthropic.com/en/docs/claude-code/subagents. Claude Code documentation page describing custom AI sub agents.

Anthropic. 2026a. Introducing claude managed agents. https://claude.com/blog/claude-managed-agents. Accessed: 2026-04-09.

Anthropic. 2026b. Sandbox-runtime: A lightweight sandboxing tool for enforcing filesystem and network restrictions on arbitrary processes at the OS level, without requiring a container. https://github.com/anthropic-experimental/sandbox-runtime. GitHub repository.

Christoph Bühler, Matteo Biagiola, Luca Di Grazia, and Guido Salvaneschi. 2025. Securing ai agent execution. *arXiv preprint arXiv:2510.21236*.

Nghi DQ Bui. 2026. Building ai coding agents for the terminal: Scaffolding, harness, context engineering, and lessons learned. *arXiv preprint arXiv:2603.05344*.

Jinkun Chen, Sher Badshah, Xuemin Yu, and Sijia Han. 2025a. Static sandboxes are inadequate: Modeling societal complexity requires open-ended co-evolution in llm-based multi-agent simulations. *arXiv preprint arXiv:2510.13982*.

Shiqi Chen, Jingze Gai, Ruochen Zhou, Jinghan Zhang, Tongyao Zhu, Junlong Li, Kangrui Wang, Zihan Wang, Zhengyu Chen, Klara Kaleb, et al. 2026a. Skillcraft: Can llm agents learn to use tools skillfully? *arXiv preprint arXiv:2603.00718*.

Tianyi Chen, Yinheng Li, Michael Solodko, Sen Wang, Nan Jiang, Tingyuan Cui, Junheng Hao, Jongwoo Ko, Sara Abdali, Leon Xu, et al. 2026b. Cua-skill: Develop skills for computer using agent. *arXiv preprint arXiv:2601.21123*.

Tianyu Chen, Dongrui Liu, Xia Hu, Jingyi Yu, and Wenjie Wang. 2026c. A trajectory-based safety audit of clawdbot (openclaw). *arXiv preprint arXiv:2602.14364*.

Yixing Chen, Yiding Wang, Siqi Zhu, Haofei Yu, Tao Feng, Muhan Zhang, Mostofa Patwary, and Jiaxuan You. 2025b. Multi-agent evolve: Llm self-improve through co-evolution. *arXiv preprint arXiv:2510.23595*.

Daixuan Cheng, Shaohan Huang, Yuxian Gu, Huatong Song, Guoxin Chen, Li Dong, Wayne Xin Zhao, Ji-Rong Wen, and Furu Wei. 2026. Llm-in-sandbox elicits general agentic intelligence. *arXiv preprint arXiv:2601.16206*.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. 2025. Mem0: Building production-ready ai agents with scalable long-term memory. *arXiv preprint arXiv:2504.19413*.

Sadia Sultana Chowa, Riasad Alvi, Subhey Sadi Rahman, Md Abdur Rahman, Mohaimenul Azam Khan Raiaan, Md Rafiqul Islam, Mukhtar Hussain, and Sami Azam. 2026. From language to action: a review of large language models as autonomous agents and tool users. *Artificial Intelligence Review*, 59(2):1–59.

João Coelho, Jingjie Ning, Jingyuan He, Kangrui Mao, Abhijay Paladugu, Pranav Setlur, Jiahe Jin, Jamie Callan, João Magalhães, Bruno Martins, et al. 2025. Deepresearchgym: A free, transparent, and reproducible evaluation sandbox for deep research. *arXiv preprint arXiv:2505.19253*.

Yuhong Dai, Jianxun Lian, Yitian Huang, Wei Zhang, Mingyang Zhou, Mingqi Wu, Xing Xie, and Hao Liao. 2025. Pretraining context compressor for large language models with embedding-based memory. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 28715–28732.

Ivan Daunis. 2025. A declarative language for building and orchestrating llm-powered agent workflows. *arXiv preprint arXiv:2512.19769*.

Xinhao Deng, Yixiang Zhang, Jiaqing Wu, Jiaqi Bai, Sibo Yi, Zhuoheng Zou, Yue Xiao, Rennai Qiu, Jianan Ma, Jialuo Chen, et al. 2026. Taming openclaw: Security analysis and mitigation of autonomous llm agent threats. *arXiv preprint arXiv:2603.11619*.

Mahir Labib Dihan, Tanzima Hashem, Mohammed Eunus Ali, and Md Rizwan Parvez. 2025. Weboperator: Action-aware tree search for autonomous agents in web environment. *arXiv preprint arXiv:2512.12692*.

Dujian Ding, Ankur Mallick, Shaokun Zhang, Chi Wang, Daniel Madrigal, Mirian Del Carmen Hipolito Garcia, Menglin Xia, Laks VS Lakshmanan, Qingyun Wu, and Victor Rühle. 2025. Best-route: Adaptive llm routing with test-time optimal compute. *arXiv preprint arXiv:2506.22716*.

Elzo Brito dos Santos Filho. 2026. Esaa: Event sourcing for autonomous agents in llm-based software engineering. *arXiv preprint arXiv:2602.23193*.

Hanxi Fang, Supawit Chockchowwat, Hari Sundaram, and Yongjoo Park. 2025a. Large-scale evaluation of notebook checkpointing with ai agents. In *Proceedings of the Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*, pages 1–8.

Tianqing Fang, Hongming Zhang, Zhisong Zhang, Kaixin Ma, Wenhao Yu, Haitao Mi, and Dong Yu. 2025b. Webevolver: Enhancing web agent self-improvement with co-evolving world model. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 8970–8986.

Tongtong Feng, Xin Wang, Zekai Zhou, Ren Wang, Yuwei Zhan, Guangyao Li, Qing Li, and Wenwu Zhu. 2025. Evoagent: Self-evolving agent with continual world model for long-horizon tasks. *arXiv preprint arXiv:2502.05907*.

Mohamed Amine Ferrag, Norbert Tihanyi, and Merouane Debbah. 2025. From llm reasoning to autonomous ai agents: A comprehensive review. *arXiv preprint arXiv:2504.19678*.

Tianyi Fu, Brian Jauw, and Mohan Sridharan. 2025. Combining llm, non-monotonic logical reasoning, and human-in-the-loop feedback in an assistive ai agent. In *2025 34th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, pages 322–329. IEEE.

Janika Deborah Gajo, Gerarld Paul Merales, Jerome Escarcha, Brenden Ashley Molina, Gian Nartea, Emmanuel G Maminta, Juan Carlos Roldan, and Rowel O Atienza. 2025. Sari sandbox: A virtual retail store environment for embodied ai agents. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2369–2378.

Yudong Gao, Zongjie Li, Zimo Ji, Pingchuan Ma, Shuai Wang, et al. 2026. Skillreducer: Optimizing llm agent skills for token efficiency. *arXiv preprint arXiv:2603.29919*.

Xinshuai Guo, Jiayi Kuang, Linyue Pan, Yinghui Li, Yangning Li, Hai-Tao Zheng, Ying Shen, Di Yin, and Xing Sun. 2026. Evoconfig: Self-evolving multi-agent systems for efficient autonomous environment configuration. *arXiv preprint arXiv:2601.16489*.

Xiyu Guo, Shan Wang, Chunfang Ji, Xuefeng Zhao, Wenhao Xi, Yaoyao Liu, Qinglan Li, Chao Deng, and Junlan Feng. 2025. Towards generalized routing: Model and agent orchestration for adaptive and efficient inference. *arXiv preprint arXiv:2509.07571*.

Bernal J Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. 2024. Hipporag: Neurobiologically inspired long-term memory for large language models. *Advances in neural information processing systems*, 37:59532–59569.

Chaoyue He, Xin Zhou, Di Wang, Hong Xu, Wei Liu, and Chunyan Miao. 2026. Harness engineering for language agents: The harness layer as control, agency, and runtime. *Preprints*. https://doi.org/10.20944/preprints202603.1756.v1.

Yufei He, Ruoyu Li, Alex Chen, Yue Liu, Yulin Chen, Yuan Sui, Cheng Chen, Yi Zhu, Luca Luo, Frank Yang, et al. 2025. Enabling self-improving agents to learn at test time with human-in-the-loop guidance. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 1625–1653.

Mengkang Hu, Tianxing Chen, Qiguang Chen, Yao Mu, Wenqi Shao, and Ping Luo. 2025a. Hiagent: Hierarchical working memory management for solving long-horizon agent tasks with large language model. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 32779–32798.

Yuanzhe Hu, Yu Wang, and Julian McAuley. 2025b. Evaluating memory in llm agents via incremental multi-turn interactions. *arXiv preprint arXiv:2507.05257*.

Naman Jain, Jaskirat Singh, Manish Shetty, Liang Zheng, Koushik Sen, and Ion Stoica. 2025. R2e-gym: Procedural environments and hybrid verifiers for scaling open-weights swe agents. *arXiv preprint arXiv:2504.07164*.

Huiqiang Jiang, Qianhui Wu, Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2024. Longllmlingua: Accelerating and enhancing llms in long context scenarios via prompt compression. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1658–1677.

Yanna Jiang, Delong Li, Haiyu Deng, Baihe Ma, Xu Wang, Qin Wang, and Guangsheng Yu. 2026. Sok: Agentic skills–beyond tool use in llm agents. *arXiv preprint arXiv:2602.20867*.

Zhengbo Jiao et al. 2026. Agentic proposing: Enhancing large language model reasoning via compositional skill synthesis. *arXiv preprint arXiv:2602.03279*.

Cheng Jiayang, Dongyu Ru, Lin Qiu, Yiyang Li, Xuezhi Cao, Yangqiu Song, and Xunliang Cai. 2026. Amemgym: Interactive memory benchmarking for assistants in long-horizon conversations. *arXiv preprint arXiv:2603.01966*.

Yihong Jin, Ze Yang, Xinhe Xu, Yihan Zhang, and Shuyang Jie. 2025. Adaptive fault tolerance mechanisms of large language models in cloud computing environments. In *2025 6th International Conference on Computer Engineering and Application (ICCEA)*, pages 754–757. IEEE.

Minki Kang, Wei-Ning Chen, Dongge Han, Huseyin A Inan, Lukas Wutschitz, Yanzhi Chen, Robert Sim, and Saravan Rajmohan. 2025. Acon: Optimizing context compression for long-horizon llm agents. *arXiv preprint arXiv:2510.00615*.

Adam Kaufman, James Lucassen, Tyler Tracy, Cody Rushing, and Aryan Bhatt. 2025. Basharena: A control setting for highly privileged ai agents. *arXiv preprint arXiv:2512.15688*.

Doyoung Kim, Zhiwei Ren, Jie Hao, Zhongkai Sun, Lichao Wang, Xiyao Ma, Zack Ye, Xu Han, Jun Yin, Heng Ji, et al. 2026. Beyond perfect apis: A comprehensive evaluation of llm agents under real-world api complexity. *arXiv preprint arXiv:2601.00268*.

LangChain. 2026a. Durable Execution in LangGraph. https://docs.langchain.com/oss/python/langgraph/durable-execution. LangChain documentation page describing durable execution in LangGraph.

LangChain. 2026b. The two patterns by which agents connect sandboxes. https://blog.langchain.com/the-two-patterns-by-which-agents-connect-sandboxes/. LangChain Blog.

Yoonho Lee, Roshen Nair, Qizheng Zhang, Kangwook Lee, Omar Khattab, and Chelsea Finn. 2026. Meta-harness: End-to-end optimization of model harnesses. *arXiv preprint arXiv:2603.28052*.

Frank Li. 2026a. Openclaw prism: A zero-fork, defense-in-depth runtime security layer for tool-augmented llm agents. *arXiv preprint arXiv:2603.11853*.

Han Li, Letian Zhu, Bohan Zhang, Rili Feng, Jiaming Wang, Yue Pan, Earl T Barr, Federica Sarro, Zhaoyang Chu, and He Ye. 2026a. Contextbench: A benchmark for context retrieval in coding agents. *arXiv preprint arXiv:2602.05892*.

Hao Li, Ruoyao Wen, Shanghao Shi, Ning Zhang, and Chaowei Xiao. 2026b. Agentdyn: A dynamic open-ended benchmark for evaluating prompt injection attacks of real-world agent security system. *arXiv preprint arXiv:2602.03117*.

Hao Li, Yankai Yang, G Edward Suh, Ning Zhang, and Chaowei Xiao. 2026c. Reasalign: Reasoning enhanced safety alignment against prompt injection attack. *arXiv preprint arXiv:2601.10173*.

Tianhao Li, Chuangxin Chu, Yujia Zheng, Bohan Zhang, Neil Zhenqiang Gong, and Chaowei Xiao. 2026d. A2asecbench: A protocol-aware security benchmark for agent-to-agent multi-agent systems. In *The Fourteenth International Conference on Learning Representations*.

Xiangyi Li, Wenbo Chen, Yimin Liu, Shenghan Zheng, Xiaokun Chen, Yifeng He, Yubo Li, Bingran You, Haotian Shen, Jiankai Sun, et al. 2026e. Skillsbench: Benchmarking how well agent skills work across diverse tasks. *arXiv preprint arXiv:2602.12670*.

Xiaoxiao Li. 2026b. When single-agent with skills replace multi-agent systems and when they fail. *arXiv preprint arXiv:2601.04748*.

Tobias Lindenbauer, Igor Slinko, Ludwig Felder, Egor Bogomolov, and Yaroslav Zharov. 2025. The complexity trap: Simple observation masking is as efficient as llm summarization for agent context management. *arXiv preprint arXiv:2508.21433*.

George Ling, Shanshan Zhong, and Richard Huang. 2026. Agent skills: A data-driven analysis of claude skills for extending large language model functionality. *arXiv preprint arXiv:2602.08004*.

Hanbing Liu, Chunhao Tian, Nan An, Ziyuan Wang, Pinyan Lu, Changyuan Yu, and Qi Qi. 2026a. Budget-constrained agentic large language models: Intention-based planning for costly tool use. *arXiv preprint arXiv:2602.11541*.

Jiaqi Liu, Yaofeng Su, Peng Xia, Siwei Han, Zeyu Zheng, Cihang Xie, Mingyu Ding, and Huaxiu Yao. 2026b. Simplemem: Efficient lifelong memory for llm agents. *arXiv preprint arXiv:2601.02553*.

Shanyv Liu, Xuyang Yuan, Tao Chen, Zijun Zhan, Zhu Han, Danyang Zheng, Weishan Zhang, and Shaohua Cao. 2026c. Caster: Breaking the cost-performance barrier in multi-agent orchestration via context-aware strategy for task efficient routing. *arXiv preprint arXiv:2601.19793*.

Songyang Liu, Chaozhuo Li, Chenxu Wang, Jinyu Hou, Zejian Chen, Litian Zhang, Zheng Liu, Qiwei Ye, Yiming Hei, Xi Zhang, et al. 2026d. Clawkeeper: Comprehensive safety protection for openclaw agents through skills, plugins, and watchers. *arXiv preprint arXiv:2603.24414*.

Xunzhuo Liu, Bowei He, Xue Liu, Andy Luo, Haichen Zhang, and Huamin Chen. 2026e. Adaptive vision-language model routing for computer use agents. *arXiv preprint arXiv:2603.12823*.

Yi Liu et al. 2026f. Agent skills in the wild: An empirical study of security vulnerabilities at scale. *arXiv preprint arXiv:2601.10338*.

Yi Liu et al. 2026g. Malicious agent skills in the wild: A large-scale security empirical study. *arXiv preprint arXiv:2602.06547*.

Ryan Lopopolo. 2026. Harness engineering: leveraging Codex in an agent-first world. https://openai.com/index/harness-engineering/. OpenAI Engineering Blog.

Yedidel Louck, Ariel Stulman, and Amit Dvir. 2025. Improving google a2a protocol: Protecting sensitive data and mitigating unintended harms in multi-agent systems. *arXiv preprint arXiv:2505.12490*.

Jiarui Lu, Thomas Holleis, Yizhe Zhang, Bernhard Aumayer, Feng Nan, Haoping Bai, Shuang Ma, Shen Ma, Mengyu Li, Guoli Yin, et al. 2025a. Toolsandbox: A stateful, conversational, interactive evaluation benchmark for llm tool use capabilities. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 1160–1183.

Miao Lu, Weiwei Sun, Weihua Du, Zhan Ling, Xuesong Yao, Kang Liu, and Jiecao Chen. 2025b. Scaling llm multi-turn rl with end-to-end summarization-based context management. *arXiv preprint arXiv:2510.06727*.

Mingfei Lu, Mengjia Wu, Feng Liu, Jiawei Xu, Weikai Li, Haoyang Wang, Zhengdong Hu, Ying Ding, Yizhou Sun, Jie Lu, et al. 2026a. Choosing how to remember: Adaptive memory structures for llm agents. *arXiv preprint arXiv:2602.14038*.

Yuxing Lu, Yucheng Hu, Xukai Zhao, and Jiuxin Cao. 2026b. Dytopo: Dynamic topology routing for multi-agent reasoning via semantic matching. *arXiv preprint arXiv:2602.06039*.

Qianqian Luo, Qingming Lin, Liuchang Xu, Sensen Wu, Ruichen Mao, Chao Wang, Hailin Feng, Bo Huang, and Zhenhong Du. 2026. Geojson agents: a multi-agent llm architecture for geospatial analysis—function calling vs. code generation. *Big Earth Data*, pages 1–55.

Siyuan Ma, Bo Gao, Xiaojun Jia, Simeng Qin, Tianlin Li, Ke Ma, Xiaoshuang Jia, Wenqi Ren, and Yang Liu. 2026a. Odar: Principled adaptive routing for llm reasoning via active inference. *arXiv preprint arXiv:2602.23681*.

Ziyu Ma, Shidong Yang, Yuxiang Ji, Xucong Wang, Yong Wang, Yiming Hu, Tongwen Huang, and Xiangxiang Chu. 2026b. Skillclaw: Let skills evolve collectively with agentic evolver. *arXiv preprint arXiv:2604.08377*.

Narek Maloyan and Dmitry Namiot. 2026. Breaking the protocol: Security analysis of the model context protocol specification and prompt injection vulnerabilities in tool-integrated llm agents. *arXiv preprint arXiv:2601.17549*.

Md Motaleb Hossen Manik and Ge Wang. 2026. Openclaw agents on moltbook: Risky instruction sharing and norm enforcement in an agent-only social network. *arXiv preprint arXiv:2602.02625*.

Luoxi Meng, Henry Feng, Ilia Shumailov, and Earlence Fernandes. 2025. cellmate: Sandboxing browser ai agents. *arXiv preprint arXiv:2512.12594*.

Mike A Merrill, Alexander G Shaw, Nicholas Carlini, Boxuan Li, Harsh Raj, Ivan Bercovich, Lin Shi, Jeong Yeon Shin, Thomas Walshe, E Kelly Buchanan, et al. 2026. Terminal-bench: Benchmarking agents on hard, realistic tasks in command line interfaces. *arXiv preprint arXiv:2601.11868*.

Zahra Moslemi, Keerthi Koneru, Yen-Ting Lee, Sheethal Kumar, and Ramesh Radhakrishnan. 2026. Polaris: Typed planning and governed execution for agentic ai in back-office automation. *arXiv preprint arXiv:2601.11816*.

João Moura. 2025. Crewai: Framework for orchestrating role-playing autonomous ai agents. https://github.com/crewaiinc/crewai. GitHub repository.

Jiayan Nan, Wenquan Ma, Wenlong Wu, and Yize Chen. 2025. Nemori: Self-organizing agent memory inspired by cognitive science. *arXiv preprint arXiv:2508.03341*.

Sriraam Natarajan, Saurabh Mathur, Sahil Sidheekh, Wolfgang Stammer, and Kristian Kersting. 2025. Human-in-the-loop or ai-in-the-loop? automate or collaborate? In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 28594–28600.

Kaiwen Ning, Jiachi Chen, Jingwen Zhang, Wei Li, Zexu Wang, Yuming Feng, Weizhe Zhang, and Zibin Zheng. 2026. Defining and detecting the defects of large language model-based autonomous agents. *IEEE Transactions on Software Engineering*, 52(3):1074–1093.

NousResearch. 2026. Hermes-agent: An agent that grows with you. https://github.com/NousResearch/hermes-agent.

OpenAI. 2024. Swarm: An educational framework exploring ergonomic, lightweight multi-agent orchestration. https://github.com/openai/swarm. GitHub repository.

OpenClaw Contributors. 2026. Openclaw — personal ai assistant. https://github.com/openclaw/openclaw.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G Patil, Ion Stoica, and Joseph E Gonzalez. 2023. Memgpt: Towards llms as operating systems. *arXiv preprint arXiv:2310.08560*.

Linyue Pan, Lexiao Zou, Shuo Guo, Jingchen Ni, and Hai-Tao Zheng. 2026. Natural-language agent harnesses. *arXiv preprint arXiv:2603.25723*.

Parallel Web Systems. 2025. What is an agent harness in the context of large-language models? https://parallel.ai/articles/what-is-an-agent-harness. Parallel Web Systems Article.

Yun Piao, Hongbo Min, Hang Su, Leilei Zhang, Lei Wang, Yue Yin, Xiao Wu, Zhejing Xu, Liwei Qu, Hang Li, et al. 2025. Agentbay: A hybrid interaction sandbox for seamless human-ai intervention in agentic systems. *arXiv preprint arXiv:2512.04367*.

Artem Pshenitsyn, Aleksandr Panov, and Alexey Skrynnik. 2026. Camar: Continuous actions multi-agent routing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 29651–29659.

Yujia Qin, Yining Ye, Junjie Fang, Haoming Wang, Shihao Liang, Shizuo Tian, Junda Zhang, Jiahao Li, Yunxin Li, Shijue Huang, et al. 2025. Ui-tars: Pioneering automated gui interaction with native agents. *arXiv preprint arXiv:2501.12326*.

Stefano Rando, Luca Romani, Alessio Sampieri, Luca Franco, John Yang, Yuta Kyuragi, Fabio Galasso, and Tatsunori Hashimoto. 2025. Longcodebench: Evaluating coding llms at 1m context windows. *arXiv preprint arXiv:2505.07897*.

Preston Rasmussen, Pavlo Paliychuk, Travis Beauvais, Jack Ryan, and Daniel Chalef. 2025. Zep: a temporal knowledge graph architecture for agent memory. *arXiv preprint arXiv:2501.13956*.

Alexander Roman and Jacob Roman. 2026. Orchestral ai: A framework for agent orchestration. *arXiv preprint arXiv:2601.02577*.

Prakash Roshan, Tina Prislan, Moumita Biswas, and Angela Lee. 2025. Semantic checkpointing for stateless llm agents in multi-tenant enterprise systems.

Philipp Schmid. 2026. The importance of Agent Harness in 2026. https://www.philschmid.de/agent-harness-2026. Philipp Schmid Blog.

David Schmotz, Sahar Abdelnabi, and Maksym Andriushchenko. 2025. Agent skills enable a new class of realistic and trivially simple prompt injections. *arXiv preprint arXiv:2510.26328*.

David Schmotz, Luca Beurer-Kellner, Sahar Abdelnabi, and Maksym Andriushchenko. 2026. Skill-inject: Measuring agent vulnerability to skill file attacks. *arXiv preprint arXiv:2602.20156*.

Shuai Shao, Qihan Ren, Chen Qian, Boyi Wei, Dadi Guo, Jingyi Yang, Xinhao Song, Linfeng Zhang, Weinan Zhang, Dongrui Liu, et al. 2025. Your agent may misevolve: Emergent risks in self-evolving llm agents. *arXiv preprint arXiv:2509.26354*.

Reshabh K Sharma. 2026. Contextcov: Deriving and enforcing executable constraints from agent instruction files. *arXiv preprint arXiv:2603.00822*.

Yuling Shi, Yichun Qian, Hongyu Zhang, Beijun Shen, and Xiaodong Gu. 2025. Longcodezip: Compress long context for code language models. *arXiv preprint arXiv:2510.00446*.

Shuang Sun, Huatong Song, Lisheng Huang, Jinhao Jiang, Ran Le, Zhihao Lv, Zongchao Chen, Yiwen Hu, Wenyang Luo, Wayne Xin Zhao, et al. 2026. Swe-world: Building software engineering agents in docker-free environments. *arXiv preprint arXiv:2602.03419*.

Weiwei Sun, Miao Lu, Zhan Ling, Kang Liu, Xuesong Yao, Yiming Yang, and Jiecao Chen. 2025a. Scaling long-horizon llm agent via context-folding. *arXiv preprint arXiv:2510.11967*.

Zeyi Sun, Ziyu Liu, Yuhang Zang, Yuhang Cao, Xiaoyi Dong, Tong Wu, Dahua Lin, and Jiaqi Wang. 2025b. SEAgent: Self-evolving computer use agent with autonomous learning from experience. *arXiv preprint arXiv:2508.04700*.

Balachandra Devarangadi Sunil, Isheeta Sinha, Piyush Maheshwari, Shantanu Todmal, Shreyan Mallik, and Shuchi Mishra. 2026. Memory poisoning attack and defense on memory based llm-agents. *arXiv preprint arXiv:2601.05504*.

Mirac Suzgun, Mert Yuksekgonul, Federico Bianchi, Dan Jurafsky, and James Zou. 2026. Dynamic cheatsheet: Test-time learning with adaptive memory. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7080–7106.

Kimi Team, Tongtong Bai, Yifan Bai, Yiping Bao, SH Cai, Yuan Cao, Y Charles, HS Che, Cheng Chen, Guanduo Chen, et al. 2026. Kimi k2. 5: Visual agentic intelligence. *arXiv preprint arXiv:2602.02276*.

Joan C Timoneda and Sebastián Vallejo Vera. 2025. Memory is all you need: Testing how model memory affects llm performance in annotation tasks. *arXiv preprint arXiv:2503.04874*.

Maximilian Puelma Touzel, Sneheel Sarangi, Gayatri Krishnakumar, Busra Tugce Gurbuz, Austin Welch, Zachary Yang, Andreea Musulan, Hao Yu, Ethan Kosak-Hine, Tom Gibbs, et al. 2025. Sandboxsocial: A sandbox for social media using multimodal ai agents. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-25*, pages 11100–11103.

Universal Tool Calling Protocol. 2026. Codemode: Plug-and-play library to enable agents to call MCP and UTCP tools via code execution. https://github.com/universal-tool-calling-protocol/code-mode. GitHub repository.

Guangya Wan, Mingyang Ling, Xiaoqi Ren, Rujun Han, Sheng Li, and Zizhao Zhang. 2025. Compass: Enhancing agent long-horizon reasoning with evolving context. *arXiv preprint arXiv:2510.08790*.

Haoming Wang, Haoyang Zou, Huatong Song, Jiazhan Feng, Junjie Fang, Junting Lu, Longxiang Liu, Qinyu Luo, Shihao Liang, Shijue Huang, et al. 2025a. Ui-tars-2 technical report: Advancing gui agent with multi-turn reinforcement learning. *arXiv preprint arXiv:2509.02544*.

Jiayu Wang, Yifei Ming, Zixuan Ke, Shafiq Joty, Aws Albarghouthi, and Frederic Sala. 2026a. Skillorchestra: Learning to route agents via skill transfer. *arXiv preprint arXiv:2602.19672*.

Jingbo Wang, Sendong Zhao, Jiatong Liu, Haochun Wang, Wanting Li, Bing Qin, and Ting Liu. 2026b. Orchestrating intelligence: Confidence-aware routing for efficient multi-agent collaboration across multi-scale models. *arXiv preprint arXiv:2601.04861*.

Jingbo Wang, Sendong Zhao, Haochun Wang, Yuzheng Fan, Lizhe Zhang, Yan Liu, and Ting Liu. 2025b. Optimal-agent-selection: State-aware routing framework for efficient multi-agent collaboration. *arXiv preprint arXiv:2511.02200*.

Jiongxiao Wang et al. 2025c. Reinforcement learning for self-improving agent with skill library. *arXiv preprint arXiv:2512.17102*.

Qianli Wang, Boyang Ma, Minghui Xu, and Yue Zhang. 2026c. When skills lie: Hidden-comment injection in llm agents. *arXiv preprint arXiv:2602.10498*.

Xilong Wang, Yinuo Liu, Zhun Wang, Dawn Song, and Neil Gong. 2026d. Websentinel: Detecting and localizing prompt injection attacks for web agents. *arXiv preprint arXiv:2602.03792*.

Yuhang Wang, Yuling Shi, Mo Yang, Rongrui Zhang, Shilin He, Heng Lian, Yuting Chen, Siyu Ye, Kai Cai, and Xiaodong Gu. 2026e. Swe-pruner: Self-adaptive context pruning for coding agents. *arXiv preprint arXiv:2601.16746*.

Yuhang Wang, Feiming Xu, Zheng Lin, Guangyu He, Yuzhe Huang, Haichang Gao, Zhenxing Niu, Shiguo Lian, and Zhaoxiang Liu. 2026f. From assistant to double agent: Formalizing and benchmarking attacks on openclaw for personalized local ai agent. *arXiv preprint arXiv:2602.08412*.

Zihan Wang, Kangrui Wang, Qineng Wang, Pingyue Zhang, Linjie Li, Zhengyuan Yang, Xing Jin, Kefan Yu, Minh Nhat Nguyen, Licheng Liu, et al. 2025d. Ragen: Understanding self-evolution in llm agents via multi-turn reinforcement learning. *arXiv preprint arXiv:2504.20073*.

Zora Zhiruo Wang, Apurva Gandhi, Graham Neubig, and Daniel Fried. 2025e. Inducing programmatic skills for agentic tasks. *arXiv preprint arXiv:2504.06821*.

Tianxin Wei, Noveen Sachdeva, Benjamin Coleman, Zhankui He, Yuanchen Bei, Xuying Ning, Mengting Ai, Yunzhe Li, Jingrui He, Ed H Chi, et al. 2025. Evo-memory: Benchmarking llm agent test-time learning with self-evolving memory. *arXiv preprint arXiv:2511.20857*.

Georg Wölflein, Dyke Ferber, Daniel Truhn, Ognjen Arandjelovic, and Jakob Nikolas Kather. 2025. Llm agents making agent tools. In *Proceedings of the*

63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 26092–26130.

Chunqiu Steven Xia, Zhe Wang, Yan Yang, Yuxiang Wei, and Lingming Zhang. 2025. Live-swe-agent: Can software engineering agents self-evolve on the fly? *arXiv preprint arXiv:2511.13646*.

Ke Xiao, Haoze Zhang, Runze Mao, Han Li, and Zhi X Chen. 2026. Towards llm-enabled autonomous combustion research: A literature-aware agent for self-corrective modeling workflows. *arXiv preprint arXiv:2601.01357*.

Yuan-An Xiao, Pengfei Gao, Chao Peng, and Yingfei Xiong. 2025. Improving the efficiency of llm agent systems through trajectory reduction. *arXiv preprint arXiv:2509.23586*.

Roy Xie, Deepak Gopinath, David Qiu, Dong Lin, Haitian Sun, Saloni Potdar, and Bhuwan Dhingra. 2026. Over-searching in search-augmented large language models. *arXiv preprint arXiv:2601.05503*.

Renjun Xu and Yang Yan. 2026. Agent skills for large language models: Architecture, acquisition, security, and the path forward. *arXiv preprint arXiv:2602.12430*.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. 2025. A-mem: Agentic memory for llm agents. *arXiv preprint arXiv:2502.12110*.

BY Yan, Chaofan Li, Hongjin Qian, Shuqi Lu, and Zheng Liu. 2025. General agentic memory via deep research. *arXiv preprint arXiv:2511.18423*.

Shuo Yang, Soyeon Caren Han, Xueqi Ma, Yan Li, Mohammad Reza Ghasemi Madani, and Eduard Hovy. 2026a. Evotool: Self-evolving tool-use policy optimization in llm agents via blame-aware mutation and diversity-aware selection. *arXiv preprint arXiv:2603.04900*.

Xianglin Yang, Yufei He, Shuo Ji, Bryan Hooi, and Jin Song Dong. 2026b. Zombie agents: Persistent control of self-evolving llm agents via self-reinforcing injections. *arXiv preprint arXiv:2602.15654*.

Yutao Yang, Junsong Li, Qianjun Pan, Bihao Zhan, Yuxuan Cai, Lin Du, Jie Zhou, Kai Chen, Qin Chen, Xin Li, et al. 2026c. Autoskill: Experience-driven lifelong learning via skill self-evolution. *arXiv preprint arXiv:2603.01145*.

Zonghao Ying, Xiao Yang, Siyang Wu, Yumeng Song, Yang Qu, Hainan Li, Tianlin Li, Jiakai Wang, Aishan Liu, and Xianglong Liu. 2026. Uncovering security threats and architecting defenses in autonomous agents: A case study of openclaw. *arXiv preprint arXiv:2603.12644*.

Justin Young. 2025. Effective harnesses for long-running agents. *Anthropic engineering note*.

Hongli Yu, Tinghong Chen, Jiangtao Feng, Jiangjie Chen, Weinan Dai, Qiying Yu, Ya-Qin Zhang, Wei-Ying Ma, Jingjing Liu, Mingxuan Wang, et al. 2025. Memagent: Reshaping long-context llm with multi-conv rl-based memory agent. *arXiv preprint arXiv:2507.02259*.

Danlong Yuan, Wei Wu, Zhengren Wang, Xueliang Zhao, Huishuai Zhang, and Dongyan Zhao. 2026a. Swe-minisandbox: Container-free reinforcement learning for building software engineering agents. *arXiv preprint arXiv:2602.11210*.

Dun Yuan, Fuyuan Lyu, Ye Yuan, Weixu Zhang, Bowei He, Jiayi Geng, Linfeng Du, Zipeng Sun, Yankai Chen, Changjiang Han, et al. 2026b. Beyond message passing: Toward semantically aligned agent communication. *arXiv preprint arXiv:2604.02369*.

Yanwei Yue, Guibin Zhang, Boyang Liu, Guancheng Wan, Kun Wang, Dawei Cheng, and Yiyan Qi. 2025. Masrouter: Learning to route llms for multi-agent systems. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15549–15572.

Kamer Ali Yuksel, Thiago Castro Ferreira, Mohamed Al-Badrashiny, and Hassan Sawaf. 2025. A multi-ai agent system for autonomous optimization of agentic ai solutions via iterative refinement and llm-driven feedback loops. In *Proceedings of the 1st Workshop for Research on Agent Language Models (REALM 2025)*, pages 52–62.

Caiqi Zhang, Menglin Xia, Xuchao Zhang, Daniel Madrigal, Ankur Mallick, Samuel Kessler, Victor Ruehle, and Saravan Rajmohan. 2026a. Budget-aware agentic routing via boundary-guided training. *arXiv preprint arXiv:2602.21227*.

Genghan Zhang, Weixin Liang, Olivia Hsu, and Kunle Olukotun. 2025a. Adaptive self-improvement llm agentic system for ml library development. *arXiv preprint arXiv:2502.02534*.

Genghan Zhang, Shaowei Zhu, Anjiang Wei, Zhenyu Song, Allen Nie, Zhen Jia, Nandita Vijaykumar, Yida Wang, and Kunle Olukotun. 2025b. Accelopt: A self-improving llm agentic system for ai accelerator kernel optimization. *arXiv preprint arXiv:2511.15915*.

Guibin Zhang, Muxin Fu, Guancheng Wan, Miao Yu, Kun Wang, and Shuicheng Yan. 2025c. G-memory: Tracing hierarchical memory for multi-agent systems. *arXiv preprint arXiv:2506.07398*.

Guibin Zhang, Haiyang Yu, Kaiming Yang, Bingli Wu, Fei Huang, Yongbin Li, and Shuicheng Yan. 2026b. Evoroute: Experience-driven self-routing llm agent systems. *arXiv preprint arXiv:2601.02695*.

Haozhen Zhang, Quanyu Long, Jianzhu Bao, Tao Feng, Weizhi Zhang, Haodong Yue, and Wenya Wang. 2026c. Memskill: Learning and evolving memory skills for self-evolving agents. *arXiv preprint arXiv:2602.02474*.

Haozhen Zhang, Haodong Yue, Tao Feng, Quanyu Long, Jianzhu Bao, Bowen Jin, Weizhi Zhang, Xiao Li, Jiaxuan You, Chengwei Qin, et al. 2026d. Learning query-aware budget-tier routing for runtime agent memory. *arXiv preprint arXiv:2602.06025*.

Jenny Zhang, Shengran Hu, Cong Lu, Robert Lange, and Jeff Clune. 2025d. Darwin godel machine: Open-ended evolution of self-improving agents. *arXiv preprint arXiv:2505.22954*.

Qizheng Zhang, Changran Hu, Shubhangi Upasani, Boyuan Ma, Fenglu Hong, Vamsidhar Kamanuru, Jay Rainton, Chen Wu, Mengmeng Ji, Hanchen Li, et al. 2025e. Agentic context engineering: Evolving contexts for self-improving language models. *arXiv preprint arXiv:2510.04618*.

Ruijia Zhang, Xinyan Zhao, Ruixiang Wang, Sigen Chen, Guibin Zhang, An Zhang, Kun Wang, and Qingsong Wen. 2026e. Safesieve: From heuristics to experience in progressive pruning for llm-based multi-agent communication. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 29892–29900.

Shengtao Zhang, Jiaqian Wang, Ruiwen Zhou, Junwei Liao, Yuchen Feng, Zhuo Li, Yujie Zheng, Weinan Zhang, Ying Wen, Zhiyu Li, et al. 2026f. Memrl: Self-evolving agents via runtime reinforcement learning on episodic memory. *arXiv preprint arXiv:2601.03192*.

Junhao Zheng, Chengming Shi, Xidi Cai, Qiuke Li, Duzhen Zhang, Chenxing Li, Dong Yu, and Qianli Ma. 2026a. Lifelong learning of large language model based agents: A roadmap. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 48(5):5552–5571.

YanZhao Zheng, ZhenTao Zhang, Chao Ma, YuanQiang Yu, JiHuan Zhu, Baohua Dong, and Hangcheng Zhu. 2026b. Skillrouter: Retrieve-and-rerank skill selection for llm agents at scale. *arXiv preprint arXiv:2603.22455*.

Yupeng Zheng, Zebin Xing, Qichao Zhang, Bu Jin, Pengfei Li, Yuhang Zheng, Zhongpu Xia, Yaran Chen, and Dongbin Zhao. 2026c. Planagent: A multi-modal large language agent for closed-loop vehicle motion planning. *IEEE Transactions on Cognitive and Developmental Systems*, pages 1–14. Early Access.

Yusheng Zheng, Yanpeng Hu, Tong Yu, and Andi Quinn. 2025. Agentsight: System-level observability for ai agents using ebpf. In *Proceedings of the 4th Workshop on Practical Adoption Challenges of ML for Systems*, pages 110–115.

Huichi Zhou, Yihang Chen, Siyuan Guo, Xue Yan, Kin Hei Lee, Zihan Wang, Ka Yiu Lee, Guchun Zhang, Kun Shao, Linyi Yang, et al. 2025a. Agentfly: Fine-tuning llm agents without fine-tuning llms. *arXiv preprint arXiv:2508.16153*.

Zijian Zhou, Ao Qu, Zhaoxuan Wu, Sunghwan Kim, Alok Prakash, Daniela Rus, Jinhua Zhao, Bryan Kian Hsiang Low, and Paul Pu Liang. 2025b. Mem1: Learning to synergize memory and reasoning for efficient long-horizon agents. *arXiv preprint arXiv:2506.15841*.

Jared Zhu, Minhao Hu, and Junde Wu. 2026a. Swe context bench: A benchmark for context learning in coding. *arXiv preprint arXiv:2602.08316*.

Qiming Zhu, Shunian Chen, Rui Yu, Zhehao Wu, and Benyou Wang. 2026b. From lossy to verified: A provenance-aware tiered memory for agents. *arXiv preprint arXiv:2602.17913*.

A Summary Tables of Agent Harness

Tables 1–6 provide a structured overview of all research works surveyed in this paper, organized by the four-layer taxonomy introduced in Section 2. Each entry is classified by its primary contribution type: *Framework* (reusable system or platform), *Method* (novel technique or algorithm), *Benchmark* (evaluation dataset or protocol), *Survey* (literature review), *Tool* (specific software artifact), or *Analysis* (empirical study or position paper).

Research Works	Meth. Fw.	Bench. Eval.	Featured Functionality	Layer Coverage			
				L1	*L2*	*L3*	*L4*
Young (Young, 2025)	✗	✗	long-running agent harness management	✓	✓	✗	✗
Pan et al. (Pan et al., 2026)	✓	✓	natural-language harness design	✓	✓	✓	✗
He et al. (He et al., 2026)	✓	✗	harness as control, agency, and runtime layer	✓	✓	✓	✓
Bui (Bui, 2026)	✓	✗	terminal coding agent scaffolding, context engineering, lessons learned	✓	✓	✓	✗
Lopopolo† (Lopopolo, 2026)	✗	✗	Codex-based harness engineering	✓	✓	✓	✗
Schmid† (Schmid, 2026)	✗	✗	agent harness importance analysis	✓	✓	✓	✓
Parallel† (Parallel Web Systems, 2025)	✗	✗	agent harness concept overview	✓	✓	✓	✓

Table 1: Agent Harness foundations: cross-layer conceptual works. †Blog/documentation.

Research Works	Domain	Meth. Fw.	Bench. Eval.	Featured Functionality	Autonomy Tier
Model & Agent Routing					
EvoRoute (Zhang et al., 2026b)	General	✓	✗	experience-driven self-routing	T2
Best-Route (Ding et al., 2025)	General	✓	✗	test-time optimal compute routing	T1
MASRouter (Yue et al., 2025)	General	✓	✗	multi-agent system routing learning	T2
Liu et al. (Liu et al., 2026e)	GUI	✓	✗	adaptive VLM routing for computer use	T3
CAMAR (Pshenitsyn et al., 2026)	General	✓	✗	continuous-action multi-agent routing	T2
SkillOrchestra (Wang et al., 2026a)	General	✓	✗	skill-transfer-based agent routing	T2
DyTopo (Lu et al., 2026b)	General	✓	✗	semantic-matching topology routing	T2
Wang et al. (Wang et al., 2026b)	General	✓	✗	confidence-aware multi-scale routing	T2
Zhang et al. (Zhang et al., 2026d)	General	✓	✗	query-aware budget-tier memory routing	T1
CASTER (Liu et al., 2026c)	General	✓	✗	context-aware task-efficient routing	T1
Zhang et al. (Zhang et al., 2026a)	General	✓	✗	boundary-guided budget-aware routing	T1
ODAR (Ma et al., 2026a)	General	✓	✗	active-inference adaptive routing	T2
Wang et al. (Wang et al., 2025b)	General	✓	✗	state-aware optimal agent selection	T2
Guo et al. (Guo et al., 2025)	General	✓	✗	generalized model-agent orchestration	T2
Multi-Agent Composition & Orchestration					
Compass (Wan et al., 2025)	General	✓	✗	evolving context for long-horizon reasoning	T3
Claude Subagents† (Anthropic, 2025b)	Code	✓	✗	custom AI subagent spawning	T3
Kimi K2.5 (Team et al., 2026)	GUI	✓	✗	visual agentic intelligence	T3
Swarm⋆ (OpenAI, 2024)	General	✓	✗	lightweight multi-agent orchestration	T2
CrewAI⋆ (Moura, 2025)	General	✓	✗	role-playing agent orchestration	T3
Daunis (Daunis, 2025)	General	✓	✗	declarative agent workflow language	T2
Orchestral AI (Roman and Roman, 2026)	General	✓	✗	general-purpose agent orchestration	T3
Autonomous Loop, Resilience & Human-in-the-Loop					
Jin et al. (Jin et al., 2025)	General	✓	✗	adaptive fault tolerance in cloud LLMs	T4
ESAA (dos Santos Filho, 2026)	General	✓	✗	event sourcing for autonomous agents	T4
Natarajan et al. (Natarajan et al., 2025)	General	✗	✗	human-in-the-loop vs AI-in-the-loop	T3
Fu et al. (Fu et al., 2025)	General	✓	✗	LLM + non-monotonic reasoning + HITL	T3
He et al. (He et al., 2025)	General	✓	✗	test-time learning with human guidance	T3
PlanAgent (Zheng et al., 2026c)	Embodied	✓	✗	closed-loop vehicle motion planning	T4
Yuksel et al. (Yuksel et al., 2025)	General	✓	✗	iterative refinement via LLM feedback	T3
Xiao et al. (Xiao et al., 2026)	Research	✓	✗	autonomous combustion research agent	T4
Ferrag et al. (Ferrag et al., 2025)	General	✗	✗	LLM reasoning to autonomous agents survey	T4

Table 2: Summary of Layer 1 (Execution & Orchestration) research works. Autonomy tiers: T1=Operator, T2=Collaborator, T3=Approver, T4=Observer. ⋆Open-source repository. †Blog/documentation.

Research Works	Domain	Meth. Fw.	Bench. Eval.	Capability			Featured Functionality
				Mem	*Cmp*	*Trj*	
Memory Systems							
Mem0 (Chhikara et al., 2025)	General	✓	✗	✓	✗	✗	scalable production long-term memory
Zep (Rasmussen et al., 2025)	General	✓	✗	✓	✗	✗	temporal knowledge graph memory
Timoneda et al. (Timoneda and Vera, 2025)	General	✓	✗	✓	✗	✗	memory effects on LLM annotation
AMemGym (Jiayang et al., 2026)	General	✗	✓	✓	✗	✗	interactive memory benchmarking
Hu et al. (Hu et al., 2025b)	General	✗	✓	✓	✗	✗	incremental multi-turn memory eval
Nemori (Nan et al., 2025)	General	✓	✗	✓	✗	✗	self-organizing cognitive memory
MemGPT (Packer et al., 2023)	General	✓	✗	✓	✓	✗	LLM as operating system with memory tiers
A-Mem (Xu et al., 2025)	General	✓	✗	✓	✗	✗	agentic self-organizing memory
MemAgent (Yu et al., 2025)	General	✓	✗	✓	✗	✗	multi-conv RL-based memory agent
G-Memory (Zhang et al., 2025c)	Multi	✓	✗	✓	✗	✓	hierarchical multi-agent memory tracing
HippoRAG (Gutiérrez et al., 2024)	General	✓	✗	✓	✗	✗	neurobiological long-term memory
SimpleMem (Liu et al., 2026b)	General	✓	✗	✓	✗	✗	efficient lifelong memory for agents
Yan et al. (Yan et al., 2025)	Research	✓	✗	✓	✗	✗	agentic memory via deep research
Lu et al. (Lu et al., 2026a)	General	✓	✗	✓	✗	✗	adaptive memory structure selection
Zhu et al. (Zhu et al., 2026b)	General	✓	✗	✓	✓	✗	provenance-aware tiered memory
Zheng et al. (Zheng et al., 2026a)	General	✗	✗	✓	✗	✗	lifelong learning roadmap for agents
Sunil et al. (Sunil et al., 2026)	Security	✓	✗	✓	✗	✗	memory poisoning attack and defense
Context Compression							
ACON (Kang et al., 2025)	General	✓	✗	✗	✓	✗	long-horizon agent context compression
LongLLMLingua (Jiang et al., 2024)	General	✓	✗	✗	✓	✗	prompt compression via token scoring
Lu et al. (Lu et al., 2025b)	General	✓	✗	✓	✓	✗	summarization-based context management
LongCodeBench (Rando et al., 2025)	Code	✗	✓	✗	✓	✗	1M-token coding evaluation
Context-Folding (Sun et al., 2025a)	General	✓	✗	✗	✓	✓	hierarchical trajectory folding
Dai et al. (Dai et al., 2025)	General	✓	✗	✗	✓	✗	embedding-based context compressor
SWE-Pruner (Wang et al., 2026e)	Code	✓	✗	✗	✓	✗	self-adaptive context pruning for code
Lindenbauer et al. (Lindenbauer et al., 2025)	General	✓	✗	✗	✓	✗	observation masking vs summarization
LongCodeZip (Shi et al., 2025)	Code	✓	✗	✗	✓	✗	long-context code compression
Mem1 (Zhou et al., 2025b)	General	✓	✗	✓	✓	✗	synergize memory and reasoning
ContextBench (Li et al., 2026a)	General	✗	✓	✗	✓	✗	context retrieval benchmarking
MemRL (Zhang et al., 2026f)	General	✓	✗	✓	✓	✗	runtime RL on episodic memory
HiAgent (Hu et al., 2025a)	General	✓	✗	✓	✓	✗	hierarchical working memory management
SWE Context Bench (Zhu et al., 2026a)	Code	✗	✓	✗	✓	✗	context learning benchmarking
SafeSieve (Zhang et al., 2026e)	Multi	✓	✗	✗	✓	✗	progressive multi-agent comm pruning
Trajectory Persistence & Observability							
Xiao et al. (Xiao et al., 2025)	General	✓	✗	✗	✓	✓	trajectory reduction for efficiency
Roshan et al. (Roshan et al., 2025)	General	✓	✗	✗	✗	✓	semantic checkpointing for stateless agents
Fang et al. (Fang et al., 2025a)	Code	✗	✓	✗	✗	✓	notebook checkpointing evaluation
AgentTrace (AlSayyad et al., 2026)	General	✓	✗	✗	✗	✓	structured logging for observability
AgentSight (Zheng et al., 2025)	General	✓	✗	✗	✗	✓	eBPF-based system-level observability
LangGraph Durable† (LangChain, 2026a)	General	✓	✗	✗	✗	✓	fault-tolerant durable execution

Table 3: Summary of Layer 2 (Context & Trajectory Management) research works—Part I: Memory (Mem), Compression (Cmp), and Trajectory Persistence & Observability (Trj). †Blog/documentation.

Research Works	Domain	Meth. Fw.	Bench. Eval.	Capability			Featured Functionality
				Evo	*Skl*	*Sec*	
Self-Evolving Architectures							
Darwin Godel Machine (Zhang et al., 2025d)	General	✓	✗	✓	✗	✗	open-ended self-improving evolution
Shao et al. (Shao et al., 2025)	General	✗	✗	✓	✗	✓	emergent risks in self-evolution
Live-SWE-Agent (Xia et al., 2025)	Code	✓	✗	✓	✗	✗	on-the-fly SWE agent self-evolution
Zhang et al. (Zhang et al., 2025e)	General	✓	✗	✓	✗	✗	evolving contexts for self-improvement
GEPA (Agrawal et al., 2025)	General	✓	✗	✓	✗	✗	reflective prompt evolution
Dynamic Cheatsheet (Suzgun et al., 2026)	General	✓	✗	✓	✗	✗	test-time learning with adaptive memory
Zhang et al. (Zhang et al., 2025a)	Code	✓	✗	✓	✗	✗	self-improvement for ML library dev
AccelOpt (Zhang et al., 2025b)	Code	✓	✗	✓	✗	✗	self-improving kernel optimization
AgentFly (Zhou et al., 2025a)	General	✓	✗	✓	✗	✗	fine-tuning agents without LLM FT
Chen et al. (Chen et al., 2025b)	General	✓	✗	✓	✗	✗	LLM self-improve through co-evolution
RAGEN (Wang et al., 2025d)	General	✓	✗	✓	✗	✗	self-evolution via multi-turn RL
Evo-Memory (Wei et al., 2025)	General	✗	✓	✓	✗	✗	benchmarking self-evolving memory
WebEvolver (Fang et al., 2025b)	Web	✓	✗	✓	✗	✗	co-evolving web world model
SEAgent (Sun et al., 2025b)	GUI	✓	✗	✓	✓	✗	self-evolving computer use agent
Almansoori et al. (Almansoori et al., 2025)	Research	✓	✓	✓	✗	✗	self-evolving clinical simulations
EvoAgent (Feng et al., 2025)	Embodied	✓	✗	✓	✗	✗	continual world model for long-horizon
Tool-R0 (Acikgoz et al., 2026)	General	✓	✗	✓	✓	✗	self-evolving tool learning from zero
EvoTool (Yang et al., 2026a)	General	✓	✗	✓	✓	✗	blame-aware tool-use optimization
AutoSkill (Yang et al., 2026c)	General	✓	✗	✓	✓	✗	experience-driven lifelong skill evolution
MemSkill (Zhang et al., 2026c)	General	✓	✗	✓	✓	✗	learning and evolving memory skills
EvoConfig (Guo et al., 2026)	Multi	✓	✗	✓	✗	✗	self-evolving multi-agent configuration
Xie et al. (Xie et al., 2026)	Web	✗	✓	✗	✗	✗	over-searching in search-augmented LLMs
Agentic Skills							
Wang et al. (Wang et al., 2025e)	General	✓	✗	✗	✓	✗	inducing programmatic skills
SkillCraft (Chen et al., 2026a)	General	✗	✓	✗	✓	✗	tool-use skill learning evaluation
SoK: Skills (Jiang et al., 2026)	General	✗	✗	✓	✓	✓	systematization of agentic skills
SkillReducer (Gao et al., 2026)	General	✓	✗	✗	✓	✗	token-efficient skill optimization
Xu et al. (Xu and Yan, 2026)	General	✗	✗	✓	✓	✓	skill architecture, acquisition, security
Ling et al. (Ling et al., 2026)	Code	✗	✗	✗	✓	✓	data-driven Claude skill analysis
SkillRouter (Zheng et al., 2026b)	General	✓	✗	✗	✓	✗	retrieve-and-rerank skill selection
Li (Li, 2026b)	General	✓	✗	✗	✓	✗	single-agent skills vs multi-agent
EvoSkill (Alzubi et al., 2026)	Multi	✓	✗	✓	✓	✗	automated multi-agent skill discovery
SkillsBench (Li et al., 2026e)	General	✗	✓	✗	✓	✗	skill benchmarking across diverse tasks
CUA-Skill (Chen et al., 2026b)	GUI	✓	✗	✗	✓	✗	skills for computer-using agents
Claude Skills† (Anthropic, 2025a)	Code	✓	✗	✗	✓	✗	agent skill platform launch
SAGE (Wang et al., 2025c)	General	✓	✗	✓	✓	✗	RL-based self-improving skill library
Jiao et al. (Jiao et al., 2026)	General	✓	✗	✗	✓	✗	compositional skill synthesis
SkillClaw (Ma et al., 2026b)	Multi	✓	✗	✓	✓	✗	collective skill evolution via cloud sharing
Skills Security							
Schmotz et al. (Schmotz et al., 2025)	Security	✗	✗	✗	✓	✓	skill-based prompt injection analysis
Liu et al. (Liu et al., 2026f)	Security	✗	✗	✗	✓	✓	skill security vulnerabilities at scale
Liu et al. (Liu et al., 2026g)	Security	✗	✗	✗	✓	✓	malicious skill detection study
Wang et al. (Wang et al., 2026c)	Security	✗	✗	✗	✓	✓	hidden-comment skill injection
Yang et al. (Yang et al., 2026b)	Security	✓	✗	✓	✓	✓	persistent control via self-reinforcing injection

Table 4: Summary of Layer 2 (Context & Trajectory Management) research works—Part II: Self-Evolving (Evo), Agentic Skills (Skl), and Skills Security (Sec). †Blog/documentation.

Research Works	Domain	Meth. Fw.	Bench. Eval.	Interaction Modality	Featured Functionality
Standardized Protocols & Interaction Surface					
Chowa et al. (Chowa et al., 2026)	General	✗	✗	API	LLM as autonomous agent review
Ning et al. (Ning et al., 2026)	General	✓	✗	API	LLM agent defect detection
Wolflein et al. (Wölflein et al., 2025)	General	✓	✗	API	agents making agent tools
UTCP Code-Mode⋆ (Universal Tool Calling Protocol, 2026)	Code	✓	✗	Protocol	MCP/UTCP via code execution
Yuan et al. (Yuan et al., 2026b)	Multi	✓	✗	A2A	semantically aligned agent communication
Louck et al. (Louck et al., 2025)	Security	✓	✗	A2A	A2A protocol sensitive data protection
A2ASecBench (Li et al., 2026d)	Security	✗	✓	A2A	protocol-aware A2A security benchmark
UI-TARS (Qin et al., 2025)	GUI	✓	✗	GUI	native automated GUI interaction
GeoJSON Agents (Luo et al., 2026)	General	✓	✗	API/Code	function calling vs code generation
Kim et al. (Kim et al., 2026)	General	✗	✓	API	real-world API complexity evaluation
Tool Use & Code Execution					
WebOperator (Dihan et al., 2025)	Web	✓	✗	GUI	action-aware web tree search
Terminal-Bench (Merrill et al., 2026)	Code	✗	✓	CLI	CLI task benchmarking
Liu et al. (Liu et al., 2026a)	General	✓	✗	API	budget-constrained tool planning
ToolSandbox (Lu et al., 2025a)	General	✗	✓	API	stateful tool-use evaluation

Table 5: Summary of Layer 3 (Interaction Surface & Execution Environment) research works. Modalities: API (function calling), CLI (terminal), GUI (visual), Code (code execution), Protocol (MCP/UTCP), A2A (agent-to-agent). ⋆Open-source repository.

Research Works	Domain	Meth. Fw.	Bench. Eval.	Enforcement			Featured Functionality
				Pre	*Run*	*Post*	
Sandboxing & Execution Environments							
LangChain† (LangChain, 2026b)	General	✗	✗	✗	✓	✗	agent-sandbox connection patterns
SWE-MiniSandbox (Yuan et al., 2026a)	Code	✓	✗	✗	✓	✗	container-free RL sandbox
UI-TARS-2 (Wang et al., 2025a)	GUI	✓	✗	✗	✓	✗	multi-turn RL GUI agent sandbox
LLM-in-Sandbox (Cheng et al., 2026)	General	✓	✗	✗	✓	✗	sandbox for agentic intelligence
Sari Sandbox (Gajo et al., 2025)	Web	✗	✓	✗	✓	✗	virtual retail store environment
SandboxSocial (Touzel et al., 2025)	Web	✗	✓	✗	✓	✗	social media simulation sandbox
Cellmate (Meng et al., 2025)	Web	✓	✗	✗	✓	✗	browser AI agent sandboxing
AgentBay (Piao et al., 2025)	General	✓	✗	✗	✓	✗	hybrid human-AI interaction sandbox
Chen et al. (Chen et al., 2025a)	General	✗	✗	✗	✓	✗	static sandboxes are inadequate
DeepResearchGym (Coelho et al., 2025)	Research	✗	✓	✗	✓	✓	reproducible deep research evaluation
SWE-World (Sun et al., 2026)	Code	✓	✗	✗	✓	✗	Docker-free SWE environments
R2E-Gym (Jain et al., 2025)	Code	✗	✓	✗	✓	✓	procedural environments with hybrid verifiers
Governance Boundaries							
POLARIS (Moslemi et al., 2026)	General	✓	✗	✓	✓	✗	typed planning and governed execution
ContextCov (Sharma, 2026)	General	✓	✗	✓	✗	✓	executable constraint enforcement
Sandbox-Runtime⋆ (Anthropic, 2026b)	Code	✓	✗	✓	✓	✗	OS-level filesystem/network sandboxing
Buhler et al. (Bühler et al., 2025)	General	✓	✗	✓	✓	✓	agent execution security analysis
BashArena (Kaufman et al., 2025)	Code	✗	✓	✗	✓	✓	highly-privileged agent control setting
Maloyan et al. (Maloyan and Namiot, 2026)	Security	✗	✗	✓	✗	✗	MCP specification security analysis
Agent Injection & Defense							
Skill-Inject (Schmotz et al., 2026)	Security	✗	✓	✓	✗	✗	skill file attack measurement
AgentDyn (Li et al., 2026b)	Security	✗	✓	✓	✓	✗	dynamic prompt injection benchmark
WebSentinel (Wang et al., 2026d)	Web	✓	✗	✗	✓	✗	web agent injection detection
ReasAlign (Li et al., 2026c)	General	✓	✗	✗	✓	✗	reasoning-enhanced injection safety
Wang et al. (Wang et al., 2026f)	Security	✗	✓	✓	✗	✗	formalizing attacks on OpenClaw
Deng et al. (Deng et al., 2026)	Security	✗	✗	✓	✓	✗	OpenClaw security analysis and mitigation
Ying et al. (Ying et al., 2026)	Security	✗	✗	✓	✗	✓	OpenClaw threat architecture and defenses
ClawKeeper (Liu et al., 2026d)	Security	✓	✗	✓	✓	✓	comprehensive safety via skills and watchers
Chen et al. (Chen et al., 2026c)	General	✗	✓	✗	✗	✓	trajectory-based safety audit
OpenClaw PRISM (Li, 2026a)	Security	✓	✗	✓	✓	✓	zero-fork defense-in-depth runtime
Manik et al. (Manik and Wang, 2026)	Security	✗	✗	✗	✓	✗	risky instruction sharing and norm enforcement

Table 6: Summary of Layer 4 (Constraints & Guardrails) research works. Enforcement timing: Pre (pre-execution validation), Run (runtime isolation), Post (post-execution audit). ⋆Open-source repository. †Blog/documentation.