Inside LLM Agents: How AI Workers Plan, Remember, and Act in the Real World

This presentation is inspired by the comprehensive survey "Large Language Model Agent: A Survey on Methodology, Applications and Challenges" (Luo et al., 2025). We'll focus on the core architecture, performance benchmarks, inherent risks, and crucial next steps in the development of LLM agents.

Presented by Manjunatha Inti

From Chatbots to Agents: Why This Shift Matters

The Limitations of Classic LLMs

Traditional LLMs like ChatGPT, Claude, and Gemini excel as sophisticated text autocomplete engines. They provide a single, direct answer and then stop, making them excellent for Q&A but less effective for complex, multi-step workflows.

The Power of LLM Agents

LLM agents introduce a revolutionary concept: wrapping models in continuous loops. They are designed to set goals, formulate plans, call external tools, evaluate results, and iterate. This cyclical approach is essential for real-world applications requiring sustained engagement, such as coding assistants, research copilots, and workflow automation bots.

What Is an LLM Agent? A Simple, Powerful Loop

An LLM agent operates on a continuous cycle, fundamentally different from a single response. This iterative process allows for dynamic interaction with its environment:

Observe

Reads user requests, analyzes conversation history, and integrates external data.

Plan

Determines the next best action or decomposes complex goals into manageable sub-tasks.

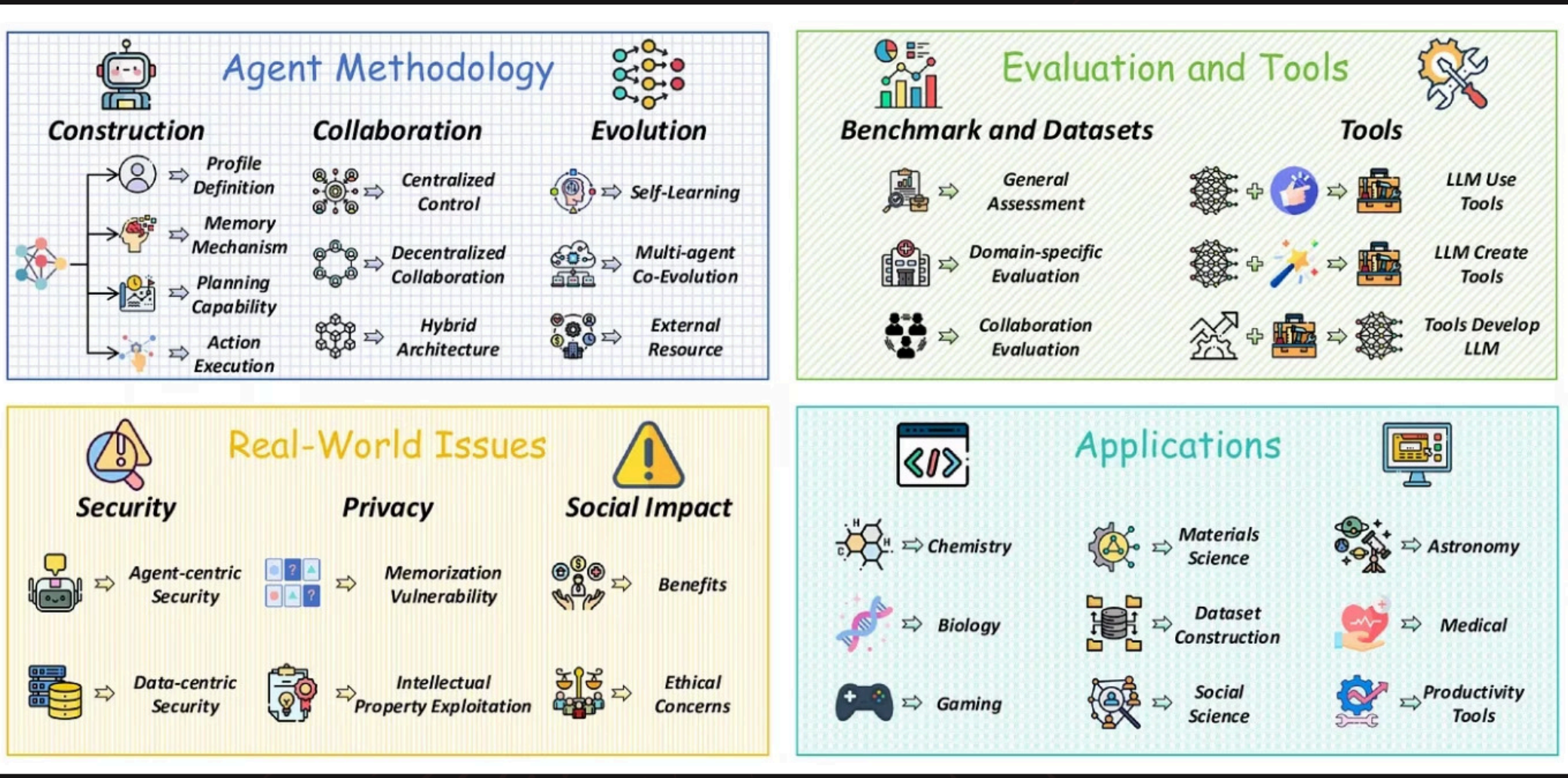
Reflect & Update

Assesses the outcome of actions, updates its internal memory, and decides whether to continue the process or stop.

Act

Executes tasks by calling various tools, including APIs, code interpreters, web browsers, databases, or even other agents.

As defined by Luo et al. (2025), agents combine an LLM with a **profile, memory, planning, and an action layer**, effectively functioning as a small "AI worker" within this loop.



Architecture at a Glance: A Comprehensive Taxonomy

The survey categorizes LLM agent methodologies into three foundational blocks, providing a structured view of their design and function:

Construction

Focuses on building individual agents, encompassing their profile definition, memory systems, planning capabilities, and tool integration.

Collaboration

Explores how multiple agents can coordinate, communicate, and share tasks to achieve collective goals more efficiently.

Evolution

Addresses how agents learn from feedback, adapt their strategies, and improve their performance over extended periods.

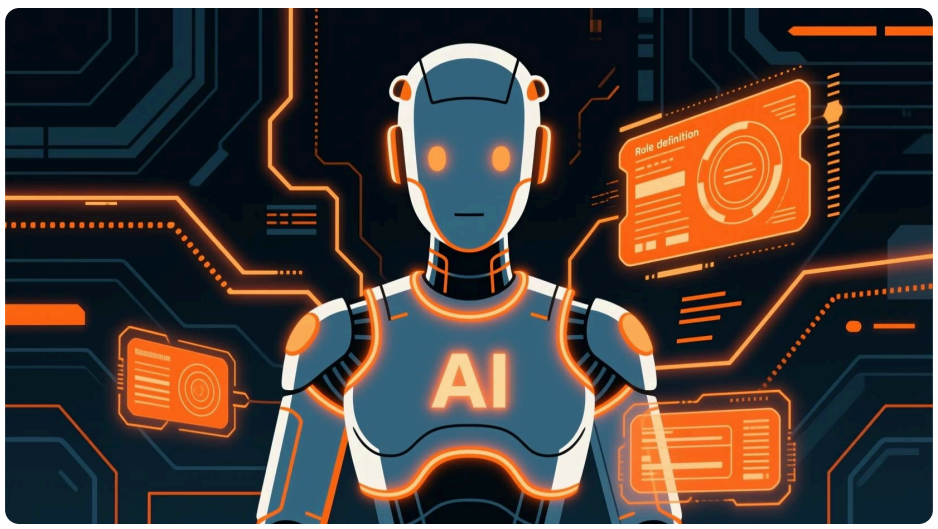
Most current systems implement only a thin slice of this architecture, typically a single role with short-term memory, a basic planner, and limited tools. A critical research question remains: which architectural blocks genuinely enhance performance versus merely adding complexity?

Inside the Agent: Profile, Memory, Planning, Tools

Each component plays a crucial role in defining an agent's capabilities and behavior:

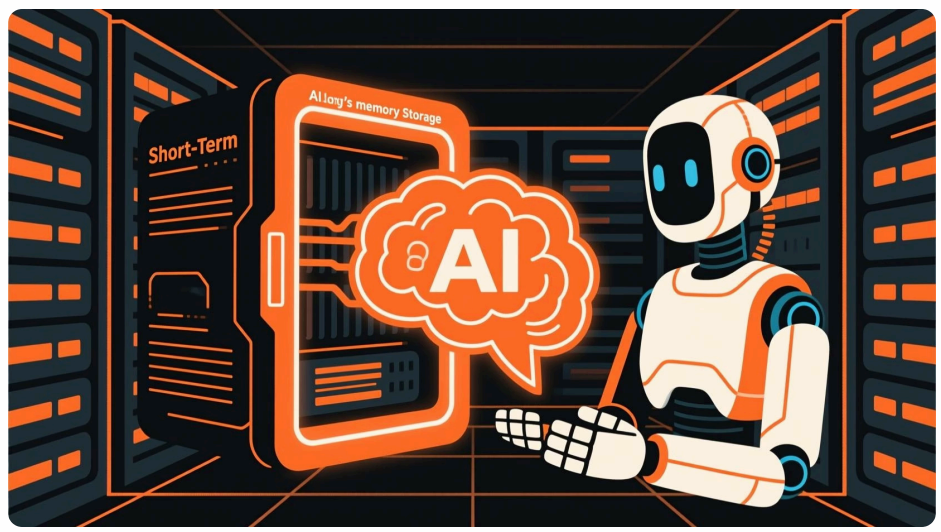
Profile

Defines the agent's core role and specific goals (e.g., "you are a data analyst" or "backend engineer"). A strong, well-defined profile leads to more focused behavior and reduces off-topic responses.



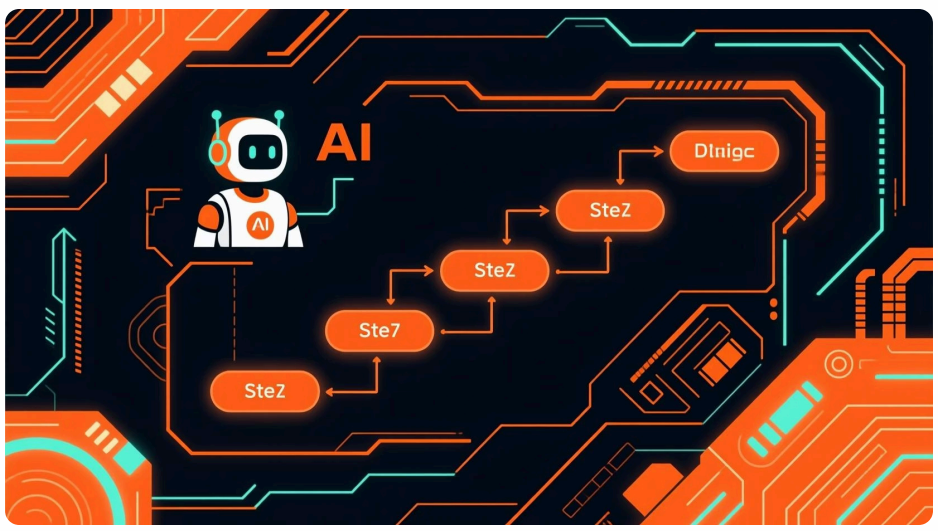
Memory

Consists of short-term memory for recent interactions and scratchpad thoughts, and long-term memory for storing skills, notes, and code snippets. Retrieval Augmented Generation (RAG) uses vector databases or knowledge graphs for factual recall.



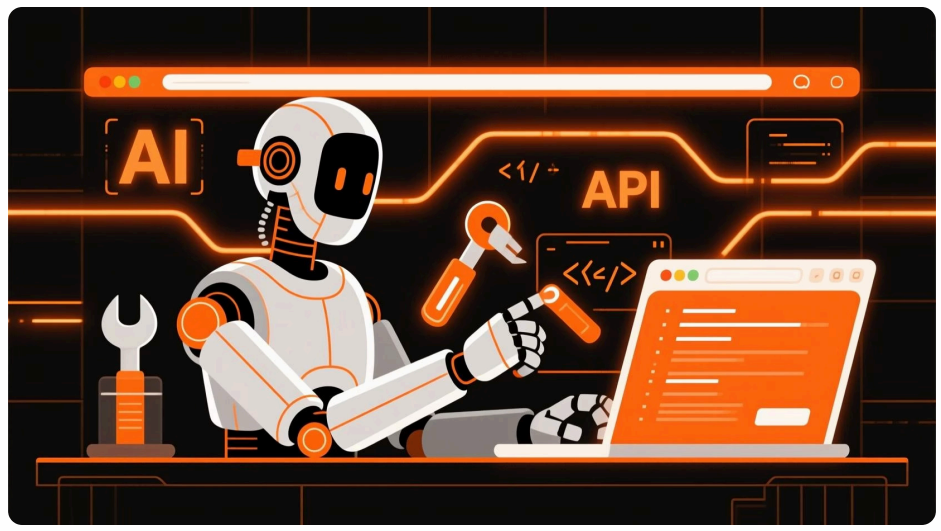
Planning

Enables the agent to break down large goals into sequential steps, explore multiple potential strategies, and refine its plans based on real-time feedback and results.



Tools / Action Layer

This layer empowers the agent to interact with the real world. It facilitates actions such as running code, calling external APIs, browsing the web, and editing documents, extending the agent's capabilities beyond mere text generation.



What the Research Says About Performance

A central question for LLM agents is whether they outperform plain LLMs on practical tasks. The survey highlights key benchmarks and evaluation metrics:

Benchmark Categories:

- **General tasks:** Multi-step reasoning and complex tool-use workflows.
- **Domain-specific tasks:** Specialized applications in coding, scientific research, medicine, gaming, and web navigation.
- **Collaboration tasks:** Multi-agent planning and negotiation scenarios.

Evaluation Metrics:

- **Task success / completion rate:** The primary measure of effectiveness.
- **Reward / score:** Quantifying performance in structured environments.
- **Human preference / trust:** Assessing the quality and reliability of outputs from a user perspective.

The emerging pattern suggests that while agents don't inherently increase the base intelligence of LLMs, they significantly enhance task completion rates, especially in lengthy and intricate workflows.

What Ablations Reveal: The Indispensable Components

Ablation studies, where specific components are removed to observe the impact, consistently reveal the critical role of each architectural piece in agent performance:

Removing Tools

Coding and web navigation tasks collapse entirely. Agents can only "talk about" actions, losing the ability to execute real-world changes.

Removing Memory

Long-duration tasks fall apart. While one-shot Q&A remains functional, multi-step tasks lose context and struggle with sequential operations.

Removing Explicit Planning

Performance on multi-step benchmarks significantly degrades. Agents that "wing it" perform demonstrably worse than those that strategize first.

My takeaway is clear: the agent's architecture is not merely cosmetic. Each component—profile, memory, planning, and tools—directly and measurably impacts task success on real-world benchmarks, underscoring their necessity for robust agent design.

Real-World Issues: Security & Privacy Challenges

While LLM agents offer immense power, they also introduce significant security and privacy vulnerabilities that must be addressed:

Security Risks

- **Prompt Injection:** Malicious instructions hidden in web pages, emails, or documents can hijack agent behavior.
- **Tool Misuse:** Agents can be coerced into data exfiltration, dangerous API calls, or executing harmful code.
- Current stress testing for jailbreaks and tool abuse remains nascent.

Privacy Risks

- Agents often handle sensitive information, including tickets, documents, logs, and production data.
- Risks of data leakage through logs, memory retention, or model memorization.
- Attacks such as membership or attribute inference become more feasible.

These challenges highlight the need for robust safeguards and ethical considerations in agent deployment.

Social Impact & My Perspective

LLM agents present a dual-edged sword, offering transformative benefits alongside significant ethical considerations:

Benefits

- Automate mundane and repetitive workflows, freeing up human potential.
- Amplify the productivity of small teams, effectively multiplying human capacity.
- Democratize access to expert tools, making complex functionalities available to non-specialists.

Ethical Concerns

- Potential for bias and unfair decisions to be scaled rapidly and widely.
- Creation of accountability gaps when agents operate autonomously.
- Disruption of labor markets, with job displacement balanced by new "agent ops" roles.

Impact	Reference
Benefits to Society	
Automation Enhancement	Foundation Models [243], GPT-3 [244], LLaMA [245]
Workforce Transformation	Foundation Models [243], Redefining Work [246]
Enhance Information Distribution	GPT-3 [244], LLaMa [245], Empower Online Education [247]
Ethical Concerns	
Bias and Discrimination	Fair Use [249], Fair Learning [250]
Accountability	Stochastic Parrots [252], Governance [253], [254]
Copyright	Fair Learning [250], Ethics of LLMs [255], AI collapse [256]
Data Privacy	Foundation Models [243], Ethical and Social Risks [257]
Manipulation & Misinformation	Data-Poisoning Attacks [259]
Others	Overreliance [244], Alignment [261], Carbon Footprint [262], Expenses [263]

Social impacts and ethical considerations of LLM agents

My take: LLM agents are powerful tools, not magical solutions. Their greatest utility lies in acting as **augmented teammates**, enhancing human capabilities rather than operating as unsupervised decision-makers.

Future Directions & Final Takeaway

The path forward for LLM agents requires a focus on reliability and responsible development, rather than a proliferation of new frameworks.

Smarter Memory

Develop mechanisms to summarize, selectively forget, and clearly separate stable knowledge from transient scratchpad information.

Safer, Structured Tool Use

Implement robust schemas, rigorous checks, and sandboxed environments to ensure secure and controlled tool execution.

Shared Benchmarks

Create comprehensive benchmarks that rigorously test planning, memory, tool integration, and, crucially, safety aspects in a holistic manner.

Better LLMOps

Establish advanced logging, continuous monitoring, and robust rollback capabilities for managing and maintaining agent behavior.

Final takeaway: LLM agents are essentially language models operating within sophisticated loops. It is these loops that truly matter. By refining the architecture, establishing clear metrics, and prioritizing safety, we can transition "AI workers" from impressive demonstrations to indispensable components of everyday workflows.

Thank You

For a deeper dive into the topics discussed, please explore the full article:

Inside LLM Agents: How AI Workers Plan, Remember, and Act in the Real World