Visualizing Deep Agents in Long-Horizon Tasks: Towards Explainable and Trustworthy Agentic AI

Amirkia Rafiei Oskooei[1,2] and **Mehmet S. Aktas**[1]
[1]Yildiz Technical University, Dept. of Computer Engineering, Istanbul, Turkey
[2]Intellica Business Intelligence Consultancy, R&D Center, Istanbul, Turkey
amirkia.oskooei@std.yildiz.edu.tr, amirkia.oskooei@intellica.net (ORCID: 0009-0004-3490-550X),
aktas@yildiz.edu.tr (ORCID: 0000-0001-7908-5067)

Abstract

The transition from prompt-based Large Language Models (LLMs) to autonomous Deep Agents has enabled the automation of long-horizon tasks. However, as these agents adopt hierarchical architectures with nested tool usage, they suffer from significant opacity. Existing linear tracing tools fail to capture the multi-dimensional complexity of parallel sub-agent execution, hindering both debugging and user trust. We propose a general-purpose observability framework that decomposes agent execution into four distinct visualization dimensions: Temporal, Cognitive, Hierarchical, and Spatial. We validate this framework through *RepoLearn*, an open-source workbench for automated codebase comprehension. Our user study demonstrates that this multi-dimensional approach reduces the Time-to-Insight (TTI) for complex behavioral analysis by 56% and significantly lowers cognitive load (NASA-TLX) compared to state-of-the-art linear traces. The source code is available at https://github.com/amirkiarafiei/repo-learn and the demo at https://www.youtube.com/watch?v=s3U6E9o94gk.

1 Introduction

The capabilities of Large Language Models (LLMs) have evolved from simple request-response to autonomous agents capable of multi-step workflows. As these systems scale to *Long-Horizon Tasks* spanning hours or days, traditional interaction paradigms break down. In automated software engineering [Deng *et al.*, 2025; Oskooei *et al.*, 2025c; Wang *et al.*, 2025; Oskooei *et al.*, 2025b; Xu *et al.*, 2024; Oskooei *et al.*, 2025a], scientific discovery [Du *et al.*, 2025; Li *et al.*, 2025; Huang *et al.*, 2025; Manus AI, 2025], or data analysis [Luo *et al.*, 2025; Ye *et al.*, 2025], supervisors can no longer monitor progress via terminal logs. The volume of intermediate steps exceeds human capacity, necessitating a shift in observability. We frame the required interface as a "Cognitive Scaffold" to track asynchronous agentic behaviors.

This capability leap is driven by a transition from standard ReAct architectures to **Deep Agents** (Agents 2.0) [Yu *et al.*, 2025; Ruan *et al.*, 2026; Huang *et al.*, 2025; Schmid, 2025; LangChain, 2025c; LangChain, 2025a]. We formally define a Deep Agent system D as a tuple characterized by four dimensions:

$$D = \{P_{temporal}, C_{cognitive}, H_{hierarchical}, E_{spatial}\} \quad (1)$$

Specifically, these systems are defined by: (1) *Sub-agents* (H), which allow for dynamic delegation and the spawning of workers with isolated contexts on-the-fly [Block, 2025; Kiro, 2025; Anthropic, 2024; Cursor, 2025]; (2) A *TODO List* (P), facilitating explicit stateful long-term planning [Erdogan *et al.*, 2025; Wang *et al.*, 2023]; (3) An *Inner Monologue* (C), separating the agent's reasoning trace from its action stream; and (4) A *Filesystem* (E), acting as permanent memory that enables the agent to offload and retrieve context [Sun *et al.*, 2025; Zhang *et al.*, 2025; Mei *et al.*, 2025]. While these features enable autonomy, they introduce significant opacity. Deep Agents currently operate as "Black Boxes." Existing observability tools [LangChain, 2025b; Xi *et al.*, 2025] offer linear trace logs, which flatten the multi-dimensional execution of parallel sub-agents into a single, overwhelming list. Debugging a recursive delegation failure or a planner hallucination using these traces is computationally challenging. Recent commercial solutions attempt to address this by employing secondary AI agents to interpret these traces, effectively using one black box to explain another. This adds latency and cost without measuring the fundamental lack of transparency. Similarly, agentic IDEs prioritize result generation over process visibility, leaving developers with a binary outcome: success or an unexplained failure. "You can't fix what you can't see."

To address this interpretability barrier, we propose a standardized, non-blocking observability framework that visualizes the four dimensions of deep agent execution: Temporal, Cognitive, Hierarchical, and Spatial. We validate this framework through *RepoLearn*, an open-source workbench for automated codebase comprehension. Our user study demonstrates that this multi-dimensional approach reduces the Time-to-Insight (TTI) for complex behavioral analysis by 56% and significantly lowers cognitive load (NASA-TLX) compared to state-of-the-art linear traces.

2 System Architecture

The system is architected as a decoupled, event-driven pipeline illustrated in Figure 1. The core engine is built upon the *Deep Agents* library [LangChain, 2026], an open-source agent harness that extends the industry-standard *LangGraph* runtime.

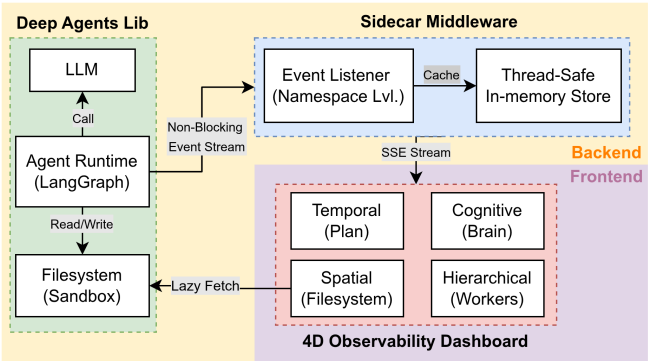

Figure 1: The System Architecture. The framework decouples the high-latency Agent Runtime (Left/Green) from the real-time Visualizer (Right/Red) via an event-driven Sidecar Middleware (Right/Blue), ensuring non-blocking observability.

This engine orchestrates the lifecycle of the Main Agent (Architect) and its dynamically spawned Sub-agents, managing their isolated context windows and tool execution.

The *Sidecar Bridge*, implemented via an asynchronous Observer pattern, operates out-of-band to intercept tool calls and state transitions without interrupting reasoning. Events are captured in real-time and buffered in a thread-safe store to ensure high-frequency updates are serialized correctly. To avoid overwhelming the client, we use a hybrid synchronization: *Transient events* (streamed thoughts and tool calls) are pushed via Server-Sent Events (SSE), while *Persistent state* (the Virtual Filesystem) uses *Event-Triggered Revalidation*. This prevents "state-bloat" and ensures high UI performance even in repositories with thousands of files. This architecture preserves agent performance while remaining non-intrusive.

3 The Observability Framework

We propose decomposing deep agent observability into four distinct dimensions (4D visualization), as in Figure 2. This framework is designed to be application-agnostic; any LangGraph-based system that exposes standard state schema keys (e.g., `plan`, `subagents`) can integrate with the visualization layer with minimal configuration.

Temporal Observability (The Plan). Visualized via the PlannerPanel (Figure 2-1), this dimension tracks the agent's explicit checklist or TODO list. Unlike linear logs, this persistent view allows users to monitor global progress, detect planner loops, and identify divergence between the intended plan and actual execution state in real-time.

Cognitive Observability (The Brain). The BrainPanel (Figure 2-3) streams the agent's "Inner Monologue" separately from its tool outputs. By isolating the reasoning trace (Thought) from execution mechanics (Action), we enable users to understand the "Why" behind behaviors without parsing verbose JSON payloads. This is critical for diagnosing hallucinations where the agent's reasoning remains sound but its actions fail.

Hierarchical Observability (The Workers). Deep agents solve complex tasks by spawning sub-agents. The WorkerGrid (Figure 2-4) employs a dynamic grid layout to visualize this delegation tree. Cards appear instantly as sub-agents are spawned, displaying their unique tool calls and status. This reveals the "hidden work" of parallel execution that is typically flattened in traditional trace views.

Spatial Observability (The Filesystem). Agents interacting with codebases modify their environment. The Virtual Filesystem (Figure 2-2) provides a real-time, read-only view of the agent's sandbox.

3.1 Comparative Analysis

Existing tooling falls into two categories: *Result-Oriented* (IDEs like Cursor, CLI tools like Claude Code) and *Trace-Oriented* (LangSmith, LangFuse). Result-oriented tools hide the process to reduce friction, leaving developers blind when reliable generation fails. Trace-oriented tools capture every event but present them as an overwhelming linear list. Our framework bridges this gap by offering *Process-Oriented* observability: it structures the data into human-readable dimensions, reducing the cognitive load required to build mental models of agent behavior.

4 Case Study & Evaluation

To validate the utility of our 4D observability framework, we integrated it into *RepoLearn*, an open-source workbench for automated codebase comprehension through a series of tutorials. Documentation of an unfamiliar repository is a representative *Long-Horizon Task*, typically requiring over 50 reasoning steps, recursive directory traversal, and the synthesis of multi-file contexts. This scenario serves as a stress test for our observability pillars, as it generates high-volume, multi-threaded execution data that is practically indecipherable in standard linear trace formats.

4.1 Study 1: Efficiency Analysis

We conducted a controlled experiment to quantify the reduction in Time-to-Insight (TTI) for complex hierarchical reasoning tasks compared to State-of-the-Art tools.

Methodology. We recruited $N = 16$ AI Engineers (Avg. 1.5 years experience; 5 female, 11 male) proficient in LangGraph-based architectures. Using a between-subjects design, participants were randomly assigned to two balanced groups ($n = 8$): one using *LangSmith* (Linear Traces) and the other using our 4D framework. Both groups engaged in the same behavioral analysis task using five recent open-source repositories (DeepAgents, TOON, Agent Lightning, POML, RAG Anything), making sure the underlying LLM we used (GPT-OSS-120B) had not seen the codebases during training. For each repository, participants answered one question from each of five observability categories: (1) *Attribution*, (2) *Causality*, (3) *Resource Usage*, (4) *Hallucination Detection*, and (5) *Planner Divergence*. This resulted in a total of 25 questions per participant. Post-task, participants completed the *NASA-TLX* survey to measure cognitive load.

Results. The results demonstrate statistically significant performance gains across both metrics.

Figure 2: The 4D Observability Dashboard. (1) **Temporal**: The Planner Panel tracks the high-level TODO list. (2) **Spatial**: The Virtual Filesystem provides a real-time view of the agent's environment. (3) **Cognitive**: The Brain Panel streams internal reasoning and action streams. (4) **Hierarchical**: The Worker Grid displays sub-agents executing scoped tasks with isolated context windows.

1. **Time-to-Insight (TTI):** The average TTI for answering the behavioral questions was reduced from **5.5 minutes** using LangSmith to **2.4 minutes** with our framework. An independent-samples t-test yielded a significant effect size (*Cohen's d* = 1.05, $p = 0.004$), indicating that the 4D decomposition effectively accelerates comprehension.
2. **Cognitive Load:** The NASA-TLX scores dropped from an average of **64.8** for the baseline to **43.7** for our framework. An independent-samples t-test confirmed a significant reduction in cognitive effort ($p < 0.001$, *Cohen's d* = 1.6). Participants reported that the ability to visually verify execution without mentally reconstructing the state from linear traces was the primary driver of this reduction.

These results support our hypothesis that 4D decomposition acts as an effective cognitive scaffold for agent supervision.

4.2 Study 2: User Trust

This study investigated whether high-fidelity process visibility (our 4D observability framework) increases user trust in autonomous agent outputs compared to standard linear traces.

Methodology. Using the same between-subjects assignment as Study 1, participants reviewed the documentation artifacts generated for the five repositories under their assigned observability paradigm:

(A) **Linear Traces**: Raw execution logs via *LangSmith*. (B) **4D Observability**: Structured execution view via *RepoLearn*. Participants rated confidence (1-5 Likert scale) on: (1) *Completeness* ("Did the agent cover relevant files?"), (2) *Accuracy* ("Did the agent hallucinate?"), and (3) *Deployability* ("Would you publish without review?").

Results. The 4D Observability condition significantly outperformed the Linear Baseline in *Accuracy* and *Deployability* ($p < 0.01$), as detailed in Table 1. Notably, while perceived *Completeness* was high across both conditions (4.0 vs 4.1), the divergence in *Deployability* (2.9 vs 3.8) underscores a critical finding: having the data (Linear) is not sufficient for trust. The structured "4D Observability" view enables users to effectively verify the agent's behavior, transforming raw information into actionable confidence.

Metric	Linear	4D	Cohen's d	p-value
Completeness	4.0	**4.1**	0.35	0.18
Accuracy	3.6	**4.3**	0.92	0.008
Deployability	2.9	**3.8**	1.05	0.003

Table 1: User Trust and Verification metrics. Scores represent average Likert ratings (1–5 scale). P-values from independent-samples t-tests (N=16, df=14, two-tailed); Cohen's d computed as standardized mean difference.

5 Conclusion

The transition to Deep Agents requires a fundamental shift from goal-oriented to process-oriented interaction. Our 4D observability framework addresses the challenge of deep agent opacity by decomposing complex behaviors into human-comprehensible temporal, cognitive, hierarchical, and spatial dimensions. Through *RepoLearn*, we demonstrated that non-blocking, multi-dimensional visualization not only reduces cognitive load by approximately 33% but also establishes the necessary trust for human-agent collaboration in high-stakes environments. Future work will explore the generalizability of this framework to other long-horizon domains, such as scientific research and automated data analysis.

Acknowledgments

This work is supported by the Scientific and Technological Research Council of Turkey (TUBITAK). The authors are grateful for the resources and opportunities provided by the council for this research.

Disclosure: Use of Generative AI

In accordance with the IJCAI-ECAI 2026 policy on Large Language Models, the authors disclose that Gemini 3.0 Pro was utilized solely for grammatical proofreading and stylistic refinement of the manuscript. The core conceptual 4D framework, system architecture of RepoLearn, experimental design, and data analysis were developed and conducted exclusively by the human authors. The authors assume full responsibility for the accuracy and integrity of the entire submission.

References

[Anthropic, 2024] Anthropic. Building agents with the Claude Agent SDK: Subagents. https://claude.com/blog/building-agents-with-the-claude-agent-sdk, 2024.

[Block, 2025] Block. Goose subagents. https://block.github.io/goose/docs/guides/subagents, 2025.

[Cursor, 2025] Cursor. Cursor subagents. https://cursor.com/docs/context/subagents, 2025.

[Deng *et al.*, 2025] Xiang Deng, Jeff Da, Edwin Pan, Yannis Yiming He, Charles Ide, Kanak Garg, Niklas Lauffer, Andrew Park, Nitin Pasari, Chetan Rane, et al. Swe-bench pro: Can ai agents solve long-horizon software engineering tasks? *arXiv preprint arXiv:2509.16941*, 2025.

[Du *et al.*, 2025] Mingxuan Du, Benfeng Xu, Chiwei Zhu, Xiaorui Wang, and Zhendong Mao. Deepresearch bench: A comprehensive benchmark for deep research agents. *arXiv preprint arXiv:2506.11763*, 2025.

[Erdogan *et al.*, 2025] Lutfi Eren Erdogan, Nicholas Lee, Sehoon Kim, Suhong Moon, Hiroki Furuta, Gopala Anumanchipalli, Kurt Keutzer, and Amir Gholami. Plan-and-act: Improving planning of agents for long-horizon tasks. *arXiv preprint arXiv:2503.09572*, 2025.

[Huang *et al.*, 2025] Yuxuan Huang, Yihang Chen, Haozheng Zhang, Kang Li, Huichi Zhou, Meng Fang, Linyi Yang, Xiaoguang Li, Lifeng Shang, Songcen Xu, et al. Deep research agents: A systematic examination and roadmap. *arXiv preprint arXiv:2506.18096*, 2025.

[Kiro, 2025] Kiro. Kiro CLI subagents. https://kiro.dev/docs/cli/chat/subagents/, 2025.

[LangChain, 2025a] LangChain. Building multi-agent applications with deep agents. https://www.blog.langchain.com/building-multi-agent-applications-with-deep-agents/, September 2025.

[LangChain, 2025b] LangChain. Debugging deep agents with langsmith. https://www.blog.langchain.com/debugging-deep-agents-with-langsmith/, December 2025.

[LangChain, 2025c] LangChain. Deep agents overview. https://www.blog.langchain.com/deep-agents/, July 2025.

[LangChain, 2026] LangChain. Deep agents: A production-ready agent harness built on LangGraph. https://github.com/langchain-ai/deepagents, 2026.

[Li *et al.*, 2025] Xiaoxi Li, Wenxiang Jiao, Jiarui Jin, Guanting Dong, Jiajie Jin, Yinuo Wang, Hao Wang, Yutao Zhu, Ji-Rong Wen, Yuan Lu, et al. Deepagent: A general reasoning agent with scalable toolsets. *arXiv preprint arXiv:2510.21618*, 2025.

[Luo *et al.*, 2025] Haotian Luo, Huaisong Zhang, Xuelin Zhang, Haoyu Wang, Zeyu Qin, Wenjie Lu, Guozheng Ma, Haiying He, Yingsha Xie, Qiyang Zhou, et al. Ultrahorizon: Benchmarking agent capabilities in ultra long-horizon scenarios. *arXiv preprint arXiv:2509.21766*, 2025.

[Manus AI, 2025] Manus AI. Manus ai: Wide research. https://manus.im/blog/introducing-wide-research, July 2025.

[Mei *et al.*, 2025] Lingrui Mei, Jiayu Yao, Yuyao Ge, Yiwei Wang, Baolong Bi, Yujun Cai, Jiazhi Liu, Mingyu Li, Zhong-Zhi Li, Duzhen Zhang, et al. A survey of context engineering for large language models. *arXiv preprint arXiv:2507.13334*, 2025.

[Oskooei *et al.*, 2025a] Amirkia Rafiei Oskooei, Kaan Baturalp Cosdan, Husamettin Isiktas, and Mehmet S Aktas. When many-shot prompting fails: An empirical study of llm code translation. *arXiv preprint arXiv:2510.16809*, 2025.

[Oskooei *et al.*, 2025b] Amirkia Rafiei Oskooei, S Selcan Yukcu, Mehmet Cevheri Bozoglan, and Mehmet S Aktas. Natural language summarization enables multi-repository bug localization by llms in microservice architectures. *arXiv preprint arXiv:2512.05908*, 2025.

[Oskooei *et al.*, 2025c] Amirkia Rafiei Oskooei, Selcan Yukcu, Mehmet Cevheri Bozoglan, and Mehmet S Aktas. Repository-level code understanding by llms via hierarchical summarization: Improving code search and bug localization. In *International Conference on Computational Science and Its Applications*, pages 88–105. Springer, 2025.

[Ruan *et al.*, 2026] Jianhao Ruan, Zhihao Xu, Yiran Peng, Fashen Ren, Zhaoyang Yu, Xinbing Liang, Jinyu Xiang, Yongru Chen, Bang Liu, Chenglin Wu, Yuyu Luo, and Jiayi Zhang. Aorchestra: Automating sub-agent creation for agentic orchestration. *arXiv preprint arXiv:2602.03786*, 2026.

[Schmid, 2025] Philipp Schmid. Agents 2.0: From shallow loops to deep agents. https://www.philschmid.de/agents-2.0-deep-agents, October 2025.

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

[Wang *et al.*, 2023] Zihao Wang, Shaofei Cai, Guanzhou Chen, Anji Liu, Xiaojian Ma, and Yitao Liang. Describe, explain, plan and select: Interactive planning with LLMs enables open-world multi-task agents. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

[Wang *et al.*, 2025] Weixuan Wang, Dongge Han, Daniel Madrigal Diaz, Jin Xu, Victor Rühle, and Saravan Rajmohan. Odysseybench: Evaluating llm agents on long-horizon complex office application workflows. *arXiv preprint arXiv:2508.09124*, 2025.

[Xi *et al.*, 2025] Yunjia Xi, Jianghao Lin, Yongzhao Xiao, Zheli Zhou, Rong Shan, Te Gao, Jiachen Zhu, Weiwen Liu, Yong Yu, and Weinan Zhang. A survey of llm-based deep search agents: Paradigm, optimization, evaluation, and challenges. *arXiv preprint arXiv:2508.05668*, 2025.

[Xu *et al.*, 2024] Frank F Xu, Yufan Song, Boxuan Li, Yuxuan Tang, Kritanjali Jain, Mengxue Bao, Zora Z Wang, Xuhui Zhou, Zhitong Guo, Murong Cao, et al. Theagentcompany: Benchmarking llm agents on consequential real world tasks. *arXiv preprint arXiv:2412.14161*, 2024.

[Ye *et al.*, 2025] Suyu Ye, Haojun Shi, Darren Shih, Hyokun Yun, Tanya Roosta, and Tianmin Shu. Realwebassist: A benchmark for long-horizon web assistance with real-world users. *arXiv preprint arXiv:2504.10445*, 2025.

[Yu *et al.*, 2025] Amy Yu, Erik Lebedev, Lincoln Everett, Xiaoxin Chen, and Terry Chen. Autonomous deep agent. *arXiv preprint arXiv:2502.07056*, 2025.

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