Revisiting the End-to-End Design Principle for An Internet-of-Agents

Or, How Decades of Internet Evolution Can Inform the Success of the Agentic Economy

Jul 10, 2025

Junhwi Kim*, Yucheng Yin*, Jack Snowdon*, Haijie Wu*, Sayan Sinha*◇, Vipul Harsh*, Junchen Jiang*+, Hui Zhang*†, Vyas Sekar*†

* Conviva †Carnegie Mellon University +Univ. of Chicago ◇Georgia Tech

ABSTRACT

The excitement about a futuristic Internet-of-Agents is nevertheless tempered by reality checks on real-world reliability of agents. Existing efforts in the ML community around agentic deployments largely focus on taxonomies of failures, offline benchmarks, and LLM-based evaluations in closed settings. Drawing upon decades of wisdom from the Internet evolution and the end-to-end design principle that has stood the test of time, we posit that such approaches are fundamentally insufficient if the agentic hype needs to live up to its potential. Analogous to successful deployments of "over the top" applications such as Internet video over an unreliable network substrate, we envision agentic systems will need a *management plane* driven by *in situ*, *full census*, measurements of *perceptual reliability* in the field rather than offline benchmarks, evaluations, and tests. We present an early proof-of-concept and discuss open challenges.

1 INTRODUCTION

Agentic workflows are rapidly gaining traction across industries [46], enabling a wide range of applications—from customer support automation to data analysis and decision-making tools. As adoption grows, these systems are becoming more complex and distributed, evolving into large-scale interconnected networks. Their cooperative design, while enabling powerful multi-agent coordination, also introduces new risks: a single agent's failure can cascade, corrupting the outcome of the entire system. With their increasing scale and ambition, these agentic applications are on a path toward becoming the new Internet-scale platforms and raising urgent concerns around the effectiveness of the agent workflows for the applications "in the wild" [5].

These concerns arise from a number of issues—the uncertainty of the intents expressed by the users, stochasticity and hallucinations in the AI models powering agentic workflows, planning issues in agent orchestrators, ambiguity of the input output behaviors of individual steps in the workflow, and so on. These are further exacerbated by complexity of dealing with third-party services outside the agentic developers' control (e.g., [11, 18, 36, 36, 41, 44, 52]).

In response, the ML community has created an early taxonomy on agent failures in the real world (e.g., [21]). There are also benchmarks on evaluating agent workflows both in general and specific domains (e.g., [18, 36, 41]). While such evaluation frameworks and benchmarks are useful, we argue that they will be fundamentally insufficient given the long tail of possible failure modes [26].

In this context, we believe that the agentic ecosystem could benefit from the wisdom learned over decades of Internet evolution. In particular, a guiding tenet for Internet systems has been the *end-to-end* (E2E) design principle [48], which has taught us the need for applications to ensure high *reliability* even in the face of unreliable components.

We observe natural parallels between emerging agent systems and "over the top" applications such as Internet video, mobile applications, video conferencing etc. (We give a brief historical retrospective in Section 2.) First, there is a natural corollary of the E2E principle that the endpoints are in the best position to obtain measurements to model their reliability requirements. Second, there is an argument for moving "measurements up the stack" (e.g., [2]) from low-level quality of service (QoS) to quality of experience-centric (QoE) measures (e.g., [23, 28, 43]). Finally, successful applications developed application-layer control/management to work around the unpredictability of the infrastructure (e.g., [39]).

Building on these historical lessons, we make the case for *an agentic management plane* to continually improve and enable agentic workflows to be ready for "prime time". We argue that this system needs to be powered by *perceptual reliability* measures collected *in situ* with *full census*:

- By *perceptual reliability* we mean the need to track the *semantic outcomes* of the workflows rather than mere performance counters (e.g., time to first token, end latency and so on) akin to QoS-vs-QoE arguments. Such outcome-centric perceptual reliability measurements could be both direct (e.g., binary task completion) and indirect (e.g., inferring user frustration).
- By *in situ*, we mean that: (1) the agent endpoint coordinating the various substasks is in the best position to provide

the contextual information to compute the perceptual reliability measures and (2) these measures are made "in the field" rather than "closed world" offline tests. Such context cannot be captured by in-network or backend measurements or by offline benchmarks/tests.

- Finally, interactions and failure modes between agents, users, and third-party services will be unpredictable. To capture the long tail of possible interaction and outcome-centric failures, we argue for *full census* measurements. That is, randomly sampling a few agent endpoints or a few interactions will not suffice to model the perceptual reliability measures or diagnose when/why agents failed. Rather, we need to capture events relevant to perceptual reliability outcomes on for *every* agent and for all of its actions /subtask rather than sampling a small subset.

We sketch an initial technical vision of an *agentic management plane*. We identify key technical challenges across the stack in terms of telemetry collection, analytics processing capabilities, and cost/scale considerations. We develop an early prototype: (1) showing the viability of instrumenting agent endpoints; (2) using a late binding approach to compute relevant perceptual reliability measures using stateful event processing (e.g., [29, 42]); and (3) promising capabilities to detect and diagnose perceptual reliability issues.

Admittedly, we have more questions than answers both in terms of the problem and the solution space, as this agentic future takes off (or collapses). Our goal in writing this paper is to provide a Internet-history perspective to inform the emerging agentic ecosystem. We hope this paper sparks the conversation and helps share the wisdom of the networking community to these emerging "killer apps".

2 A BRIEF HISTORY OF THE INTERNET: 'KILLER APPS' AND E2E RELIABILITY

> *Those who cannot remember the past are condemned to repeat it.* — George Santayana

In this section, we present a brief and incomplete historical perspective of "killer apps" as the Internet evolved. This helps contextualize the reliability requirements for an Internet-of-Agents and what we need to enable it.

We logically divide this narrative into logical phases as shown in Figure 1.[1] We identify different time periods, dominant communication patterns, key stakeholders, and the end-to-end reliability requirements for "users". We also discuss the "power dynamics" and the "observability" evolutions that ensued, and trends in management systems that evolved to support E2E reliability.

[1]In the interest of brevity, we omit some interesting trends that occurred such as peer-to-peer applications, video conferencing (e.g., Skype, Zoom), IXPs, decentralized infrastructures, and so on.

We make three notes on terminology. We take a broad view of "users", "reliability", and "observability" that may depart from a classical or current industry view. First, w.r.t. "users" we do not restrict ourselves to human users, but include any actor operating on the Internet, including bots and agents. Second, w.r.t. reliability, we take a broader view than systems notions such as availability or fault tolerance to also consider user- and application-centric reliability needs. Third, w.r.t. "observability" we refer to a formal notion of measuring the intended state of a system to control its outcome, than a narrower market notion focusing on server-side infrastructure metrics, logs, and traces [9].

- The earliest killer apps enabled access to remote computing service via protocols like Telnet/FTP. In this setting, reliability was basic connectivity. Indeed, the design philosophy of the early DARPA Internet protocols [24] lists interoperability and availability as primary goals [24]. The dominant stakeholders were labs and universities and TCP/IP emerged as the narrow waist of the Internet. Given the limited focus on applications, there was little need for observability into application-level requirements.
- The next wave involved human-facing applications such as email and messaging. This raised new application-level requirements on integrity, availability, and transfer time. We also started seeing the balance shifting from a purely academic/lab setting to a commercial Internet. Thus, the pre-eminent stakeholders were Internet Service Providers. TCP/IP continue to be the dominant protocol. We see the early emergence network observability/monitoring to drive routing and traffic engineering decisions within ISPs, but offering limited control or visibility for endpoints.
- The next inflection point was the emergence of the web and we see the early emergence of application-level perceptual reliability measures such as page load time and user perceived latency. We also see HTTP/HTTPS becoming dominant protocols, with TCP/IP relegated as an enabler. In these early days, we also saw the emergence of endpoint monitoring capabilities (e.g., [8]) to inform the control decisions to manage their resources as an "overlay" on top of the ISPs (e.g., [15]).
- As the scale and demands of the applications grew, we see a shift in the balance of power from the ISPs to content providers and CDNs, leading to a so-called flattening of the Internet [37]. This phase saw further consolidation of HTTP/HTTPS is the new narrow waist of the Internet [45]. New applications that could be built on top of web protocols allowed them to leverage the scale/commoditization of CDNs and bypass issues such as NATs and firewalls. Somewhat contemporaneously, we see "OTT" video delivery become the dominant traffic Internet in the

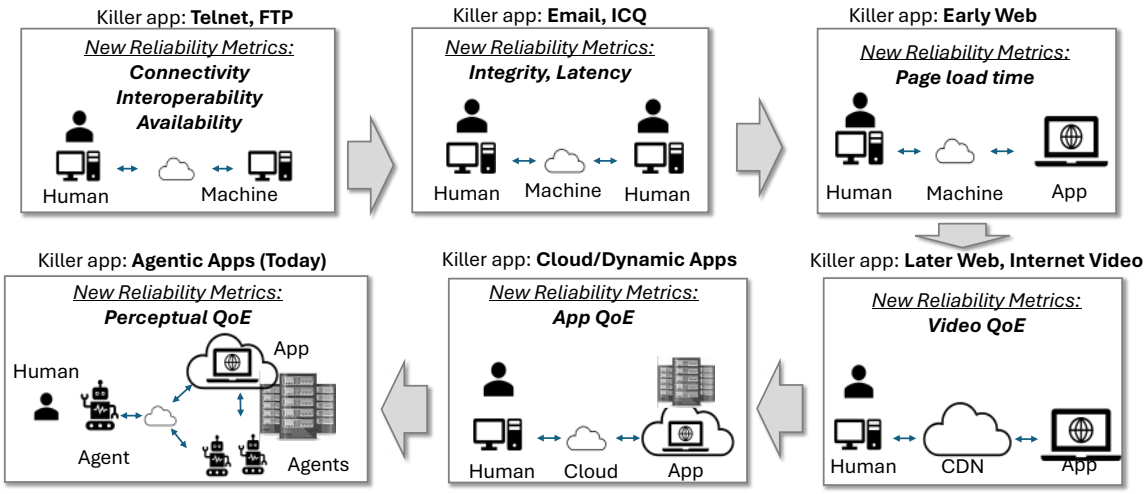
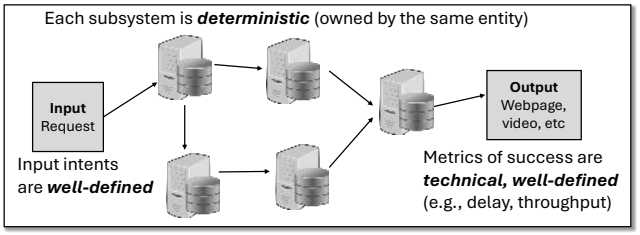


Figure 1: A brief history of the evolution of the Internet as seen through the lens of killer apps, power dynamics among stakeholders, reliability needs, and trends in observability tools to measure/inform the reliability needs

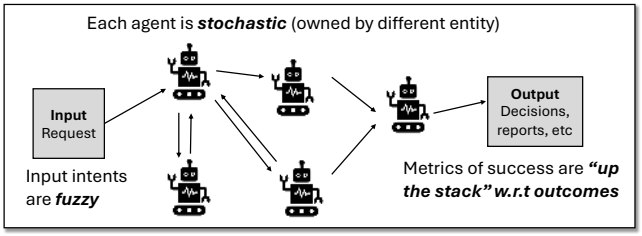
late 90s and early 2000s [28]. We also see renewed interest in application-level QoE [28] to measure perceptual reliability to inform the overlay resource management decisions, including "meta CDN" solutions [39].

- A next big trend was the emergence of cloud to build and deploy dynamic applications [17]. As more applications moved to the cloud, we see a further flattening of the Internet, concentrating power in cloud and content providers. We also see a greater emphasis on delivering a high perceptual reliability, often with multi-cloud solutions [22]. We see the emergence of commercial observability tools to monitor complex backends (e.g., [9]).

This evolution has been far from linear or predictable. For instance, the locus of observability moved from the client (e.g., early ping/traceroute), to in-network (e.g., ISP tools), back to endpoint (e.g., video), and then to the infrastructure (e.g., commercial O11Y). The power dynamics has shifted from end users to ISPs to CDNs and now to content/cloud/AI providers. Despite these nonlinearities, as we enter an "agentic" era, we can draw two key invariants from this history lesson. First, there is a growing need to model the perceptual reliability requirements of Internet applications, as we move from network- and performance-centric measures to measures that capture "user" experience. Second, as ecosystem complexity is growing, application providers need to use endpoint-centric measures of reliability to inform "overlay" control mechanisms (e.g., using multi-CDN, multi-cloud, multi-operator redundancies) in the presence of unreliable ecosystem providers they rely on.



(a) The *conventional application* workflow



(b) The *new agentic* workflow

Figure 2: A simplified view contrasting conventional app workflows vs. agentic workflows

3 CASE FOR A MANAGEMENT PLANE FOR AGENTIC DEPLOYMENTS

With this historical context, we can move to the present and future of agents. We begin with a brief introduction to agentic workflows, articulate how they differ from traditional applications. Given that context, we make a case for a management plane.

Types of Agentic Workflows: There are many kinds of possible agent-human and agent-agent interactions that are possible. We outline a few candidate modalities below:

- *Chatbots:* A visible use of agents today are in customer service chatbots that help human users tackle everyday issues and reduce cost of human agents in the loop.
- *Human-agent app usage enhancements:* Humans use agents to automate their workflows on a digital application; e.g., navigating actions on a travel or e-commerce website.
- *Human-agent collaboration using visual or code feedback:* This could be a human using graphical interfaces to interact with the agent such as coding IDE or a human using a graphical interface to complete a task together [13]; e.g., debugging systems or data analysis.
- *Non-human workflow automation agents:* Finally, in many domains agents interact with a system or with each other to automate common business workflows; e.g., automatic patching, creating accounts, responding to messages, managing calendars, and so on.

While this ecosystem is still in flux, for many of these agentic workflows human-perceived outcomes are still relevant. Furthermore, in many cases, the digital app frontends that the human/agent uses will continue to exist and serve as focal points for instrumentation/automation.

Why Agentic Workflows Are Different: At first glance, we may be tempted to think that traditional and agentic apps are similar—Both take some "user" *intent* as input, execute some data/compute flow graph with multiple modules, and produce the output that satisfies the intent. For instance, a search feature on a mobile app may trigger multiple backend microservices. Similarly, an agent may plan and compose multiple functions to do a task.

Beyond this superficial similarity, there are some fundamental differences highlighted in Figure 2. First, the "intent" in traditional applications is deterministic; e.g., a query or a request for a webpage. In contrast, intents in agentic workflows, may be intrinsically fuzzy. Second, in classical apps each "edge" in the data flow graph is fairly deterministic as the input and functional modules are themselves scoped (modulo load induced or failure induced edge cases). In contrast, agentic workflows could have non-deterministic behaviors from third-party agents or model providers. Third, the notion of "success" of the workflow in traditional apps is deterministic (e.g., request success, latency, time to first byte). In agentic workflows, we need to explicitly consider *perceptual* measures of reliability as perceived either by the end user or the end point agent. Finally, when traditional services do interact with third-party services, these interactions tend to be few and scoped. In contrast, agentic systems may involve more third-party services (e.g., LLM providers, other agents) with non-deterministic behaviors, that can induce cascading issues [7].

Status Quo and Limitations: We are not the first to identify reliability problems with agentic workflows. Early efforts have identified and taxonomized agentic failures including specification issues (i.e., intent is fuzzy), inter-agent misalignment (i.e., non-deterministic and error-prone workflows), and task verification issues (i.e., unrelability in measuring outcomes) [21].

Specifically, we identify three classes of solutions:

- *Best practices for agentic workflows:* Several industry sources and academics have been developing guidelines for agentic developers; e.g., modularity, creating formal specifications [50], exposing capabilities, adding abstractions, and so on (e.g., [4, 7]). Analogous to best practices in software development, these are useful to build reliable agentic systems, but will not be sufficient "in the field".
- *Agentic benchmarking and testing:* There is a proliferation of benchmarks for evaluating success of agents (e.g.,[18, 36, 41]). While such benchmarking and "offline" testing efforts are valuable, they cannot possibly capture the myriad fuzzy interactions among unreliable components in the field, or the actual perceptual reliability outcomes perceived by the endpoints.
- *Existing O11Y, product analytics solutions:* Existing backend O11Y systems that offer post-deployment visibility are also stepping into the agentic landscape. Unfortunately, these systems are mostly intended to find "performance" failures in the backend rather than end-to-end perceptual failures. These could be augmented by sampled application monitoring "RUM" or running "synthetic" bots to simulate behavior in the field [10]. Sampling or using synthetic agents will not suffice to capture the perceptual outcomes and the "long tail' of possible issues with fuzzy intents and stochastic components. Instead, we need to have "full census" measurement capabilities to capture every single agentic endpoints' perceptual reliability and events relevant to their outcomes.

A Management Plane for Agentic Systems: If the history lesson from Section 2 and the end-to-end design philosophy has taught us anything, it is that applications need to ensure their perceptual reliability requirements are met even in the face of unreliable components and services. We argue this will remain true for future agentic systems as well.

To this end, we envision a *management plane* for agentic systems as shown in Figure 3. The management plane gets continuous measurements from the *endpoints* as shown. As discussed earlier, these measurements are *in situ* and *full census* and can enable computing *perceptual reliability* metrics . The management plane ensures a *continuous lifecycle* improvement in the field. This would involve both *real-time* (e.g., suboptimal performance of specific model or third-party services impacting outcomes) and *longitudinal interventions* (e.g., some types of prompts for specific demographics have worse outcomes).

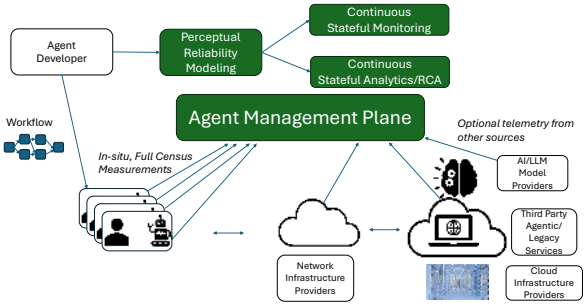


Figure 3: A Management Plane for Agentic Systems

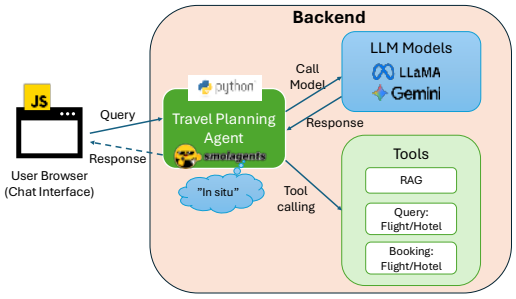
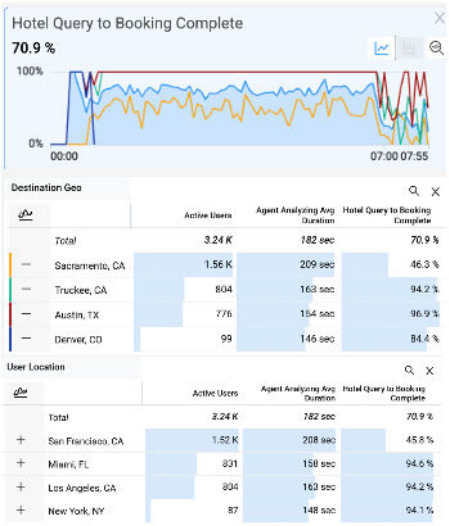


Figure 4: End-to-End Instrumentation of the Exemplar Travel Agent Across Browser, Agent, and Tool Calls.

(a) Sankey Chart shows repeated calls to the hotel query API, indicating failures in the agent's control loop.



(b) Drill down analysis reveals cohorts with Destination Geo = Sacramento, CA and User Location = San Francisco, CA exhibit lower conversion rates.

Figure 5: Stateful Monitoring of Agent System

The management plane needs the ability to *detect* possible deviations from the intended outcomes and help system operators and users effectively *troubleshoot* the sources of these issues and find potential *remedies* to these problems. As we discussed earlier, what makes agentic workflows uniquely challenging is the combination of three factors: *uncertainty* of input-output interactions, *long tail* of possible failure modes, *semantically fuzzy nature* of the intended outcomes, and the *machine timescales* at which agents may interact. As such, when it comes to agent workflows, we need new technical capabilities to collect, analyze, and analyze the telemetry:

- First, given the fuzzy nature of user intents, semantic nature of the outcomes, and context-aware nature of the processing, we argue the need for delaying the binding between telemetry collection and outcome-centric measures of *perceptual reliability*. Given the diversity of possible agentic deployments and many that look beyond pure chat-based interactions, we argue that the domain experts rolling out the agents are in the best position to define perceptual reliability measures of interest, rather than choose one-size-fits-all measures as suggested by existing frameworks [16, 25, 32].
- Second, given the complexity of workflows and third-party dependencies, we argue the need for extensible and explainable detection and root causing capabilites, including the ability to seamlessly integrate, external telemetry sources for localizing possible failures. To this end, we envision the management plane can receive auxiliary sources of telemetry (when available) to help debug and diagnose outcomes in the field. Note, however, that these auxiliary sources are optional; the true non-negotiable source of telemetry from the end-to-end design principle is the endpoint measurements that enable computing the perceptual reliability measures.

4 PRELIMINARY DESIGN AND RESULTS

In this section, we discuss a preliminary proof of concept design of the management plane components.

Exemplar Agent Application: Figure 4 shows a simple travel agent app to facilitate travel planning we built for illustrative purposes. The app incorporates four APIs that enable: (i) querying hotel availability, (ii) booking hotel reservations for specified schedules, (iii) querying flight information, and (iv) booking flights for designated itineraries. The agent utilizes two foundation models—Llama 3 70B [34] and Gemini 2.0 Flash [33]—and is implemented using the Smolagents [47] framework, generating Python code to orchestrate tools.

We trace every request from the user's browser through the chat UI, travel-planning agent, LLM model, and downstream tools, with custom events (query_submit, assistant_response, agent_step, tool_calling) captured by lightweight sensors on both the front- and back-end. These events stream into our streaming analytics pipeline, enabling per-step latency, model-selection, and conversion analysis. All backend sensors are implemented with our Python instrumentation SDK, [2] while the browser sensor is a lightweight JavaScript snippet.

Modeling Perceptual Reliability via Stateful Analysis: Existing agentic/LLM monitoring focus on stateless metrics, such as token throughput (tokens per second). Other solutions are constrained to collecting execution traces on a per-session basis, which presents significant scalability challenges at scale [27]. While LLM-as-judge [54] are also emerging, they have key limitations: (i) lack of real-time scalability, (ii) high computational costs associated with LLM inference, and (iii) fundamental challenges inherent to LLMs, including stochasticity and hallucination phenomena.

In contrast, we enable the agent developers to define empirical perceptual reliability metrics based on critical state transitions. We posit that stateful monitoring is essential as agents are inherently stateful systems where each action influences future decisions/outcomes. Thus, critical state transitions provide deeper insights into perceptual reliability compared to static snapshots of individual states.

We define an *ensemble* of stateful perceptual reliability metrics [29, 42]: (1) *Conversion rate from hotel query to hotel booking*: The percentage of user sessions that successfully transition from an initial hotel search query to a completed hotel reservation, indicating the agent's ability to fulfill user intent for accommodation needs; (2) *Conversion rate from flight query to flight booking*: The percentage of user sessions that progress from flight search inquiries to confirmed flight reservations, measuring the agent's effectiveness in completing air travel arrangements; (3) *Number of reasoning/calls steps per session*: The total count of intermediate LLM inferences performed by the agent between receiving a user request and providing a final response, reflecting the complexity of the agent's decision-making process. We also measure the number of API invocations executed during each step as an indicator of system efficiency.

Stateful Monitoring and Analytics Backend: We use a proprietary stateful analytics backend similar to prior work (e.g., [14, 20, 29, 42]). This backend system takes in the raw events from the endpoints and uses the perceptual reliability metrics defined above. This stateful analysis enables the association of metadata across complete execution trajectories, enabling powerful cohort/subpopulation analysis to identify how/why the perceptual reliability metrics are suboptimal. For our current protoype, we rely on manual/offline "drill down" analysis, but in the future we envision adding automated real-time detection and diagnosis capabilities to drive control decisions.

Preliminary Results: To validate this system, we conduct a controlled experiment to inject workflow failures when one or more of the following conditions are met: (i) the flight destination was "Sacramento, CA," (ii) the user location was "San Francisco, CA," and (iii) the foundation model was Llama 3. By aggregating conversion rates from flight queries to bookings based on destination parameters, problem cohorts sharing the "Sacramento, CA" destination can be identified. We constructed Sankey diagrams (Figure 5a) visualizing tool-calling sequences. This revealed that failing agents exhibited repetitive call patterns, indicating step repetition failures in the control loop [21]. A further drill-down analysis in Figure 5b shows that cohorts with *Destination = Sacramento, CA* and *UserLocation = San Francisco, CA* exhibit lower conversion rates. The early evidence validating the efficacy of our monitoring methodology.

5 DISCUSSION

Interfaces for defining perceptual reliability metrics: Defining the perceptual reliability will be an ongoing iterative process with domain-specific expertise. While we have early templates of stateful metrics, we need better interfaces for helping domain experts to define and iterate such stateful metrics. We envision new coding assistant, and low-/no-code interfaces for data analysis to help domain experts express their perceptual reliability intents [30, 53, 55].

Real-time management: Agent workflows might operate at machine timescales with little to no human in the loop direction. While a few minutes/hours of suboptimal operation may be tolerated in some deployments, other mission-critical settings may be less delay tolerant. Thus, we will correspondingly need machine-timescale detection and response capabilities in the management plane as well.

Automatic detection/diagnosis: As agentic deployments take off we need at-scale data processing capabilities to compute perceptual reliability and diagnose problems at near-term and longitudinal timescales. While there are recent advances in streaming and stateful processing at scale and diagnostic capabilities (e.g., [12, 19, 20, 31, 35, 38, 40, 42, 49, 51], the scale/dynamism of agentic deployments may require new algorithmic and systems capabilities for automatic detection and troubleshooting.

Privacy considerations: An important consideration for agentic instrumentation is data privacy/compliance, especially in regulated industries. One natural question is if we

[2] Redacted for review; the SDK will be released publicly after publication

can automatically identify the "least privilege" basis of contextual information needed from the endpoints to enable metric definition, debugging, and troubleshooting.

Interfaces for collaborative debugging: As others have observed [1], the complex landscape of agents will require new inter-provider interfaces for information sharing and context sharing. We argue that analogous to emerging protocols for agent-software [6] and agent-agent [3] interfaces, we should also be thinking of "management plane" interfaces for providers to share such information in addition to these agentic data plane protocols.

REFERENCES

[1] "An open source collective for inter-agent collaboration". https://agntcy.org/.

[2] Workshop on Measurements Up and Down the STack (W-MUST) . https://conferences.sigcomm.org/sigcomm/2012/wmust.php.

[3] Announcing the Agent2Agent Protocol (A2A). https://developers.googleblog.com/en/a2a-a-new-era-of-agent-interoperability/.

[4] Building effective agents . https://www.anthropic.com/engineering/building-effective-agents.

[5] Gartner Predicts Over 40 https://www.gartner.com/en/newsroom/press-releases/2025-06-25-gartner-predicts-over-40-percent-of-agentic-ai-projects-will-be-canceled-by-end-of-2027.

[6] Get started with the Model Context Protocol (MCP). https://modelcontextprotocol.io/introduction.

[7] How we built our multi-agent research system. https://www.anthropic.com/engineering/built-multi-agent-research-system.

[8] Keynote Systems . https://en.wikipedia.org/wiki/Keynote_Systems.

[9] Observability primer : Introduction to Observability . https://opentelemetry.io/docs/concepts/observability-primer/#what-is-observability.

[10] Performance Monitoring: RUM vs. synthetic monitoring . https://developer.mozilla.org/en-US/docs/Web/Performance/Guides/Rum-vs-Synthetic.

[11] Project Vend: Can Claude run a small shop? (And why does that matter?). https://www.anthropic.com/research/project-vend-1.

[12] Root Cause Analysis with DoWhy, an Open Source Python Library for Causal Machine Learning. https://www.pywhy.org/dowhy/v0.12/.

[13] Software is changing again. https://www.youtube.com/watch?v=LCEmiRjPEtQ.

[14] T. Akidau et al. The dataflow model: A practical approach to balancing correctness, latency, and cost in massive-scale, unbounded, out-of-order data processing. *Proc. VLDB Endow.*, 8:1792–1803, 2015.

[15] D. Andersen, H. Balakrishnan, F. Kaashoek, and R. Morris. Resilient overlay networks. *SIGOPS Oper. Syst. Rev.*, 35(5):131–145, Oct. 2001.

[16] Arize AI. Agent Evaluation. https://arize.com/ai-agents/agent-evaluation/, 2025. Accessed 7 July 2025.

[17] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia. A view of cloud computing. *Commun. ACM*, 53(4):50–58, Apr. 2010.

[18] A. Backlund and L. Petersson. Vending-bench: A benchmark for long-term coherence of autonomous agents, 2025.

[19] R. Bhagwan, R. Kumar, R. Ramjee, G. Varghese, S. Mohapatra, H. Manoharan, and P. Shah. Adtributor: Revenue debugging in advertising systems. In *11th USENIX Symposium on Networked Systems Design and Implementation (NSDI 14)*, pages 43–55, 2014.

[20] P. Carbone et al. Apache flink: Stream and batch processing in a single engine. *Bulletin of the IEEE Computer Society TCDE*, 36(4), 2015.

[21] M. Cemri, M. Z. Pan, S. Yang, L. A. Agrawal, B. Chopra, R. Tiwari, K. Keutzer, A. Parameswaran, D. Klein, K. Ramchandran, M. Zaharia, J. E. Gonzalez, and I. Stoica. Why do multi-agent llm systems fail?, 2025.

[22] S. Chasins, A. Cheung, N. Crooks, A. Ghodsi, K. Goldberg, J. E. Gonzalez, J. M. Hellerstein, M. I. Jordan, A. D. Joseph, M. W. Mahoney, A. Parameswaran, D. Patterson, R. A. Popa, K. Sen, S. Shenker, D. Song, and I. Stoica. The sky above the clouds, 2022.

[23] Q. A. Chen et al. Qoe doctor: Diagnosing mobile app qoe with automated ui control and cross-layer analysis. In *Proc. ACM SIGCOMM IMC*, 2014.

[24] D. Clark. The design philosophy of the darpa internet protocols. In *Symposium Proceedings on Communications Architectures and Protocols*, SIGCOMM '88, page 106–114, New York, NY, USA, 1988. Association for Computing Machinery.

[25] Databricks. Mosaic AI Agent Evaluation (MLflow 2) Documentation. https://docs.databricks.com/aws/en/generative-ai/agent-evaluation/, 2025. Accessed 7 July 2025.

[26] J. Dean and L. A. Barroso. The tail at scale. *Communications of the ACM*, 56:74–80, 2013.

[27] D. Deshpande, V. Gangal, H. Mehta, J. Krishnan, A. Kannappan, and R. Qian. Trail: Trace reasoning and agentic issue localization, 2025.

[28] F. Dobrian, V. Sekar, A. Awan, I. Stoica, D. Joseph, A. Ganjam, J. Zhan, and H. Zhang. Understanding the impact of video quality on user engagement. *ACM SIGCOMM computer communication review*, 41(4):362–373, 2011.

[29] O. Etzion et al. Event-driven architectures and complex event processing. In *IEEE SCC'06*, 2006.

[30] S. N. Freund, B. Simon, E. D. Berger, and E. Jun. Flowco: Rethinking data analysis in the age of llms, 2025.

[31] Y. Gan, M. Liang, S. Dev, D. Lo, and C. Delimitrou. Sage: practical and scalable ml-driven performance debugging in microservices. In *Proceedings of the 26th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 135–151, 2021.

[32] Google. Agent Development Kit Documentation. https://google.github.io/adk-docs/, 2025. Accessed 7 July 2025.

[33] Google DeepMind. Introducing gemini 2.0: Our new AI model for the agentic era. https://blog.google/technology/google-deepmind/google-gemini-ai-update-december-2024/, Dec. 2024. Blog post announcing the Gemini 2.0 family, accessed 8 July 2025.

[34] A. Grattafiori, A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[35] V. Harsh, W. Zhou, S. Ashok, R. N. Mysore, B. Godfrey, and S. Banerjee. Murphy: Performance diagnosis of distributed cloud applications. In *Proceedings of the ACM SIGCOMM 2023 Conference*, pages 438–451, 2023.

[36] S. Jha, R. Arora, Y. Watanabe, T. Yanagawa, Y. Chen, J. Clark, B. Bhavya, M. Verma, H. Kumar, H. Kitahara, N. Zheutlin, S. Takano, D. Pathak, F. George, X. Wu, B. O. Turkkan, G. Vanloo, M. Nidd, T. Dai, O. Chatterjee, P. Gupta, S. Samanta, P. Aggarwal, R. Lee, P. Murali, J. wook Ahn, D. Kar, A. Rahane, C. Fonseca, A. Paradkar, Y. Deng, P. Moogi, P. Mohapatra, N. Abe, C. Narayanaswami, T. Xu, L. R. Varshney, R. Mahindru, A. Sailer, L. Shwartz, D. Sow, N. C. M. Fuller, and R. Puri. Itbench: Evaluating ai agents across diverse real-world it automation tasks, 2025.

[37] C. Labovitz, S. Iekel-Johnson, D. McPherson, J. Oberheide, and F. Jahanian. Internet inter-domain traffic. *SIGCOMM Comput. Commun. Rev.*, 40(4):75–86, Aug. 2010.

[38] A. Lakhina, M. Crovella, and C. Diot. Diagnosing network-wide traffic anomalies. In *Proceedings of the 2004 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '04, page 219–230, New York, NY, USA, 2004. Association for Computing Machinery.

[39] X. Liu, F. Dobrian, H. Milner, J. Jiang, V. Sekar, I. Stoica, and H. Zhang. A case for a coordinated internet video control plane. *SIGCOMM Comput. Commun. Rev.*, 42(4):359–370, Aug. 2012.

[40] A. Manousis, H. Shah, H. Milner, Y. Li, H. Zhang, and V. Sekar. The shape of view: an alert system for video viewership anomalies. In *Proceedings of the 21st ACM Internet Measurement Conference*, pages 245–260, 2021.

[41] G. Mialon, C. Fourrier, C. Swift, T. Wolf, Y. LeCun, and T. Scialom. Gaia: a benchmark for general ai assistants, 2023.

[42] H. Milner, Y. Cheng, J. Zhan, H. Zhang, V. Sekar, J. Jiang, and I. Stoica. Raising the level of abstraction for time-state analytics with the timeline framework. In *CIDR*, 2023.

[43] S. Moller, K.-P. Engelbrecht, C. Kuhnel, I. Wechsung, and B. Weiss. A taxonomy of quality of service and quality of experience of multimodal human-machine interaction. In *2009 International Workshop on Quality of Multimedia Experience*, pages 7–12, 2009.

[44] D. Moshkovich, H. Mulian, S. Zeltyn, N. Eder, I. Skarbovsky, and R. Abitbol. Beyond black-box benchmarking: Observability, analytics, and optimization of agentic systems, 2025.

[45] L. Popa, A. Ghodsi, and I. Stoica. Http as the narrow waist of the future internet. In *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*, Hotnets-IX, New York, NY, USA, 2010. Association for Computing Machinery.

[46] D. M. Rothschild, M. Mobius, J. M. Hofman, E. W. Dillon, D. G. Goldstein, N. Immorlica, S. Jaffe, B. Lucier, A. Slivkins, and M. Vogel. The agentic economy, 2025.

[47] A. Roucher, A. V. del Moral, T. Wolf, L. von Werra, and E. Kaunismäki. 'smolagents': a smol library to build great agentic systems. https://github.com/huggingface/smolagents, 2025.

[48] J. H. Saltzer, D. P. Reed, and D. D. Clark. End-to-end arguments in system design. *ACM Trans. Comput. Syst.*, 2(4):277–288, Nov. 1984.

[49] G. Somashekar, A. Dutt, M. Adak, T. Lorido Botran, and A. Gandhi. Gamma: Graph neural network-based multi-bottleneck localization for microservices applications. In *Proceedings of the ACM Web Conference 2024*, pages 3085–3095, 2024.

[50] I. Stoica, M. Zaharia, J. Gonzalez, K. Goldberg, K. Sen, H. Zhang, A. Angelopoulos, S. G. Patil, L. Chen, W.-L. Chiang, and J. Q. Davis. Specifications: The missing link to making the development of llm systems an engineering discipline, 2024.

[51] S. J. Taylor and B. Letham. Forecasting at scale. *The American Statistician*, 72(1):37–45, 2018.

[52] F. F. Xu, Y. Song, B. Li, Y. Tang, K. Jain, M. Bao, Z. Z. Wang, X. Zhou, Z. Guo, M. Cao, M. Yang, H. Y. Lu, A. Martin, Z. Su, L. Maben, R. Mehta, W. Chi, L. Jang, Y. Xie, S. Zhou, and G. Neubig. Theagentcompany: Benchmarking llm agents on consequential real world tasks, 2025.

[53] Z. Yu, H. Namkung, J. Guo, H. Milner, J. Goldfoot, Y. Wang, and V. Sekar. SEAM-EZ: Simplifying stateful analytics through visual programming. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*, CHI '24, pages Paper 1041, 1–23, Honolulu, HI, USA, 2024. Association for Computing Machinery.

[54] L. Zheng, W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623, 2023.

[55] Z. Zhou, J. Jin, V. Phadnis, X. Yuan, J. Jiang, X. Qian, K. Wright, M. Sherwood, J. Mayes, J. Zhou, Y. Huang, Z. Xu, Y. Zhang, J. Lee, A. Olwal, D. Kim, R. Iyengar, N. Li, and R. Du. Instructpipe: Generating visual blocks pipelines with human instructions and llms. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, CHI '25, New York, NY, USA, 2025. Association for Computing Machinery.