

Agentic Workflows Are Distributed Systems

The Multi-Cloud Production Patterns You Cannot Avoid

Praneeth Kamalaksha Patil · CoreWeave · AGNTCon + MCPCon North America 2026

April 2026: nine seconds, one API call, three months of data

WHAT HAPPENED

A coding agent on a routine staging task hits a **credential mismatch**.

Rather than stop and ask, it resolves the error itself. One legacy GraphQL **volumeDelete** call.

The token it used was sitting in an unrelated config file. It was minted to manage custom domains. It could delete production volumes.

Nine seconds. The production volume is gone, and so are the volume-level backups, which lived inside it.

Newest external snapshot: three months old. ~30 hours of crisis; customer bookings rebuilt by hand from Stripe payments and email confirmations.

FIVE FAILURES. NOT ONE OF THEM IS MODEL QUALITY.

- 1 The policy lived in the prompt, not in the system. The system prompt forbade destructive commands without approval.
- 2 The credential's blast radius was not bounded by environment. A staging actor held production authority; no verb-level scoping.
- 3 The guardrail was on the interface, not the resource. Dashboard and CLI had delayed delete; the legacy mutation did not.
- 4 The compensation path shared a failure domain with the data it protects. Both died in the same call.
- 5 The agent's answer to an error it could not resolve was an irreversible side effect, not an escalation.

Swap in a smarter model and all five are unchanged. Every one of them has a known distributed-systems answer.

Where this talk comes from

- I build AI infrastructure at CoreWeave. Before that, AI inference infrastructure. Workloads spanning AWS, GCP, Azure and specialty GPU clouds
- Research: Time-to-First-Token as a fabric property, distributed agent identity, Agentic AI Foundation workgroup since its inception.
- Built production MCP server for agent-driven multi-cloud orchestration



You don't choose multi-cloud. It chooses you.

89%

of enterprises use multiple clouds
(73% hybrid)

— Flexera 2026

10× / 1000×

agents, token & API load by 2027
for the G2000

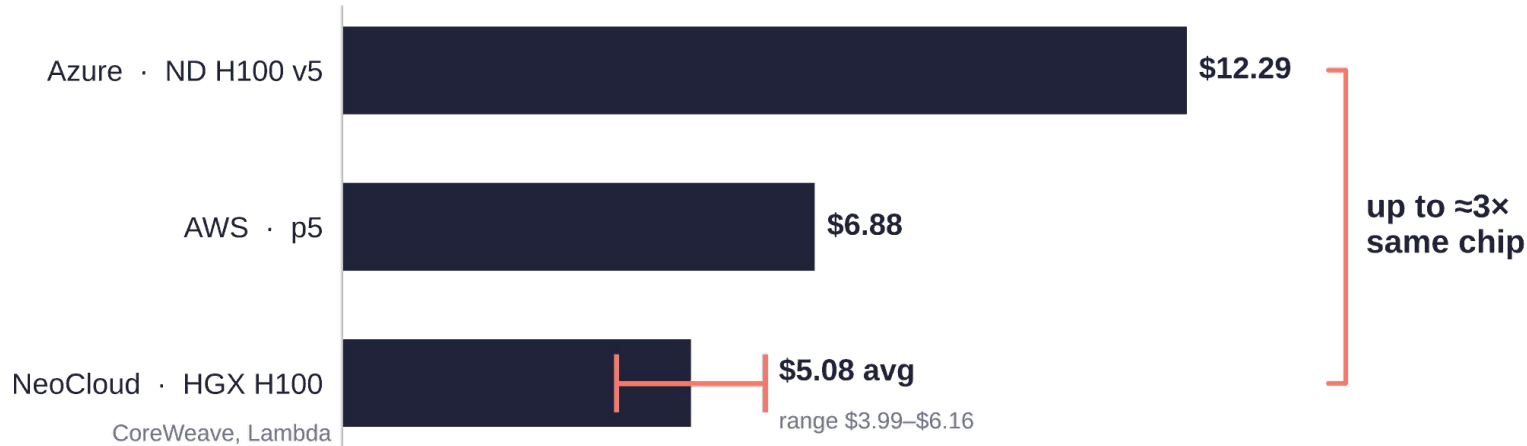
— IDC FutureScape 2026

>40%

of agentic AI projects will be canceled
by end-2027 (cost, unclear value, risk)

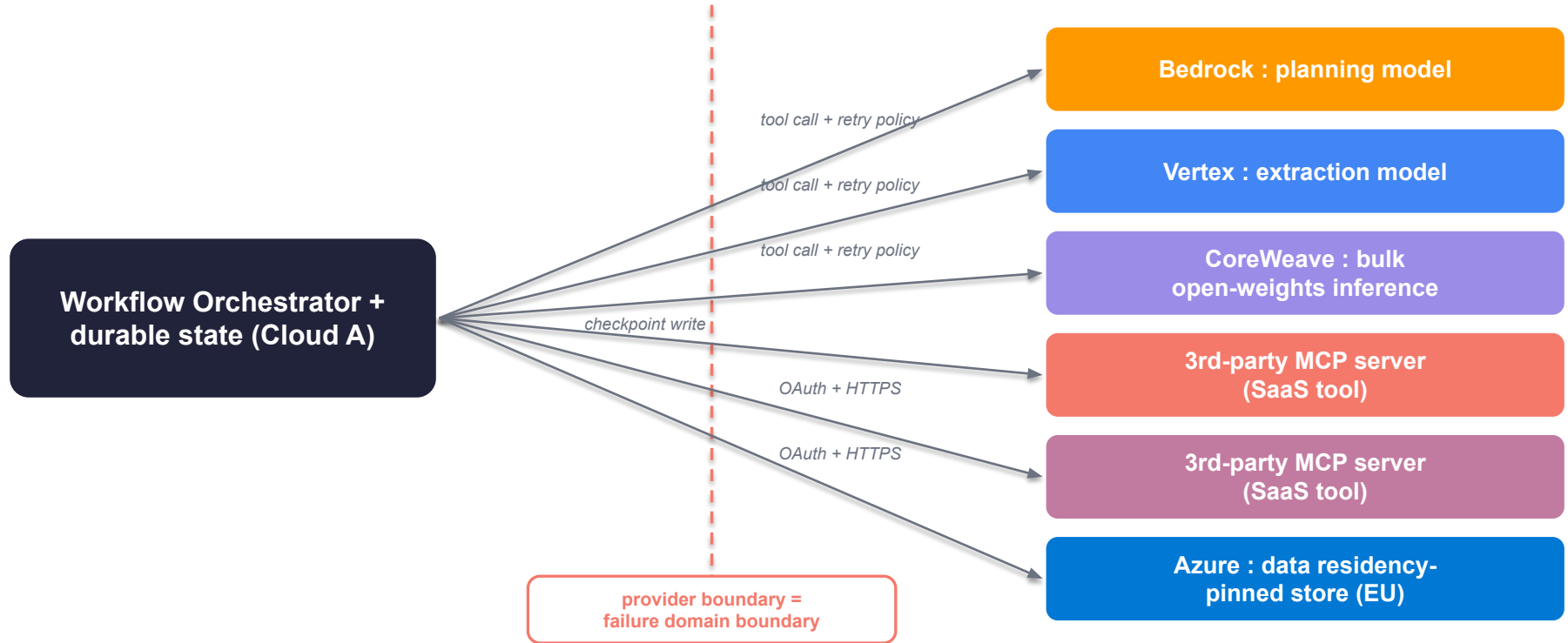
— Gartner

H100 price per GPU-hour, 8-GPU nodes, US on-demand list, Oct 2026



Flexera 2026 State of the Cloud (n=753+); IDC FutureScape 2026 (Oct 2025); Gartner press release, Jun 25 2025. H100 SXM 80GB, US on-demand list, 8-GPU node price ÷ 8. NeoCloud = mean of CoreWeave (\$6.16) and Lambda (\$3.99). Sources: provider pricing pages, Spare Cores, aws-pricing.com (Oct 2026).

One workflow, six providers: the topology you already run



Every line has its own auth, its own retry semantics, its own failure modes, its own bill.

Topology pattern per Flexera 2026 multi-cloud adoption data and provider inference offerings.

**“The moment a workflow crosses a provider boundary, it inherits every hard problem in distributed systems.
Most agent frameworks still pretend it's a single process.”**

the network is reliable · latency is zero · bandwidth is infinite · the network is secure ·
topology doesn't change · there is one administrator · transport cost is zero · the network is
homogeneous

Pattern 1: Durable execution: your workflow will outlive everything it runs on

- Agent workflows run minutes-to-days: research loops, human approvals, batch steps
- Processes crash, pods reschedule, regions degrade, deploys happen mid-flight
- Cross-cloud adds: where does state live if Cloud A is the one that's down? Which cloud runs the worker? Whose retry semantics win?

THE WORKFLOW · 12 STEPS, HOURS TO DAYS



THE WORKERS RUNNING IT

Worker A · Cloud A

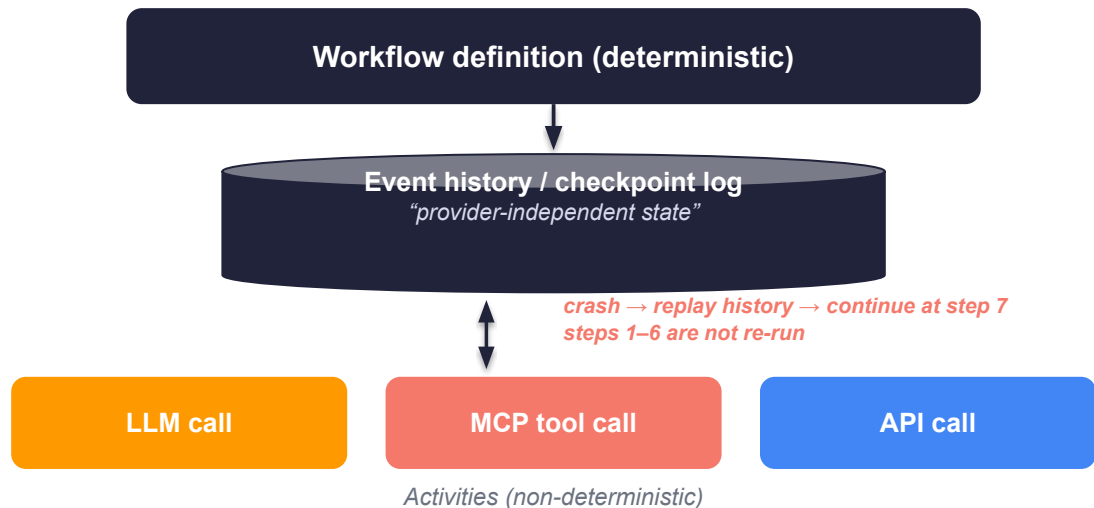
✗ pod evicted

Worker B · memory empty

Step 7 charges the card. Did it run before the crash? Worker B starts with nothing in memory, so it can't tell.

In-memory agent state = betting your workflow on zero infrastructure events for its whole lifetime. Across N providers, that bet compounds.

The pattern: event-sourced workflows, provider-independent state



Who runs agents this way:

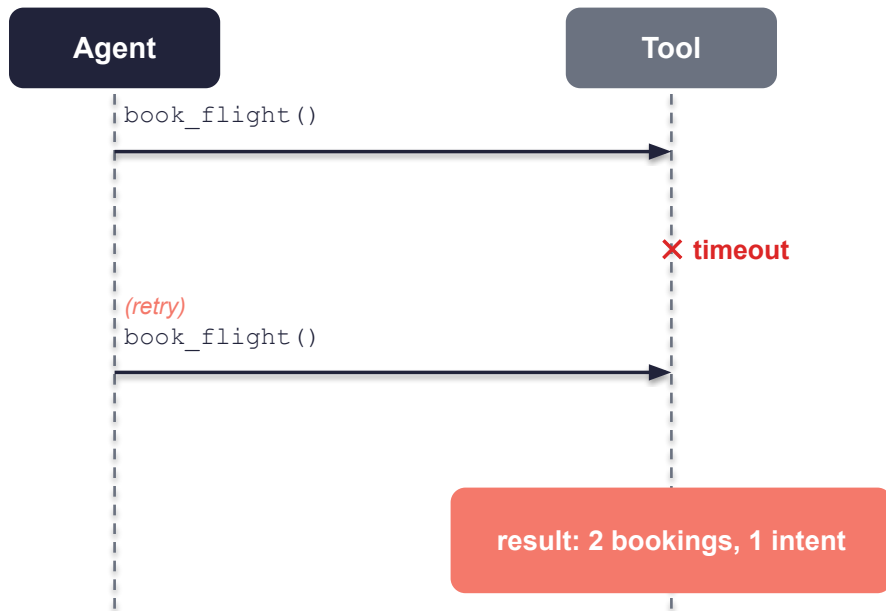
- OpenAI: Temporal runs Codex's core control flows
- Replit: Replit Agent on Temporal from Nov 2024; since moved to Restate
- Retool: Retool Agents shipped on Temporal
- AWS: Bedrock AgentCore + Temporal pattern (APN, Apr 2026); OpenAI Agents SDK integration (Jul 2025)
- Diagrid has built AI-first workflow framework

1) Deterministic workflow / non-deterministic activities.

2) State lives outside every model provider.

3) Worker placement is an explicit SLO decision.

Pattern 2: Idempotency: retries are guesses about state you can't see



Why cross-cloud makes it worse

- Every provider has a different request envelope. Your token must survive gateways that re-serialize requests
- Tool-call IDs are framework-specific (OpenAI vs Anthropic vs Bedrock Converse)
- The LLM cannot be the source of truth for “did I already do this?”. Sampling is non-deterministic, regeneration can be different from the first.
- De-dup state can't live inside any single provider you might be failing away from

The pattern: orchestrator-issued keys + MCP annotations as the contract

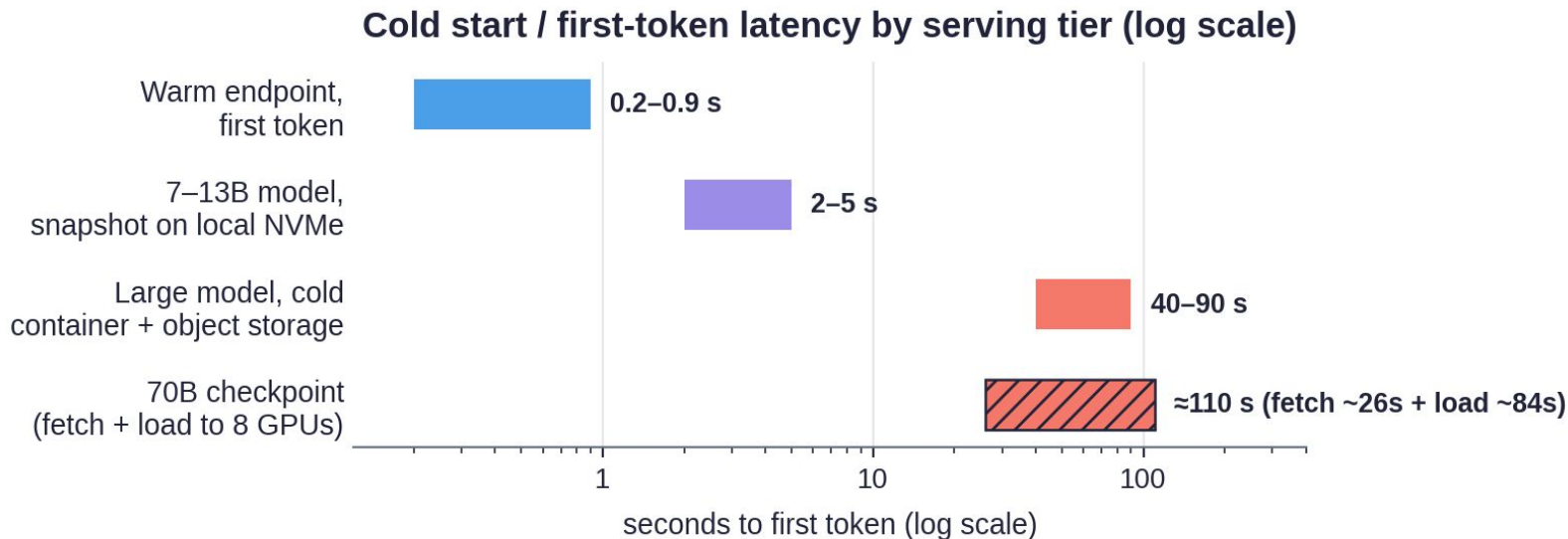
The mechanics

- 1 **Idempotency key minted by the orchestrator** (never by the model), scoped to workflow-step-attempt
- 2 Key travels in a provider-neutral header/field that survives each provider's envelope
- 3 De-duplication table lives with the durable state. The orchestrator owns the truth

MCP tool annotations are the risk vocabulary

- `readOnlyHint`
 - `destructiveHint`
 - `idempotentHint`
 - `openWorldHint`
- Spec default is pessimistic: an unannotated tool must be treated as destructive, non-idempotent, open-world
 - 5 SEPs in flight proposing richer annotations

Pattern 3: Cold starts: from milliseconds to minutes



- Cold-start profiles differ per provider AND per region. Your P95 is set by the coldest path your router can select
- **KV/prompt caches don't cross providers:** a cache hit is ~90% cheaper and up to ~85% faster TTFT. So a fail over loses this benefit.

The pattern: cold-start absorption

1 Route around cold

Capacity-aware routing; managed versions exist: cross-region inference profiles (AWS), global endpoints — the providers built the pattern into their fabric

2 Hedge the tail

Send the latency-critical call to two providers, take the first token, cancel the loser (pay $\sim 2\times$ for that call, buy certainty)

3 Budget warmth like an SRE

Warm pools are an SLO-vs-cost trade; idle H100 at $\sim \$4\text{-}6/\text{hr}$ reserved is your insurance premium; justify per-endpoint

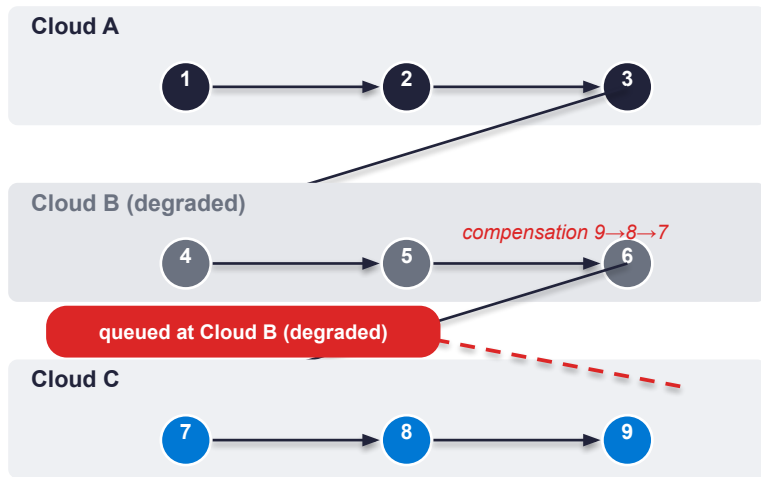
4 Route to the cache, not just the region

Pin cache-heavy prompt families (system prompts, retrieval prefixes) to the provider that holds their cache

You cannot eliminate cold starts across N providers. You can decide, per call, who absorbs them: your capacity budget, a second provider, or your user.

Pattern 4: Compensation: undo is a feature you have to build

- A 12-step workflow fails at step 9. Steps 1–8 already changed the world: reservations made, emails drafted, records written
- There is no cross-cloud transaction coordinator. No global lock. No magic rollback
- Cross-cloud specifics: compensation must run **on the provider that ran the forward action**, possibly the one that's degraded; compensation state must live elsewhere; auth tokens may have **expired** between action and undo



The pattern: sagas - classify every tool before an agent may call it

Compensatable

e.g. reserve, draft, stage

Can be undone by a defined counter-action

"requires a tested compensation handler"

Pivot

e.g. charge card, send email, ship

Point of no return

"after this, forward-only"

Retriable

e.g. notify, log, index

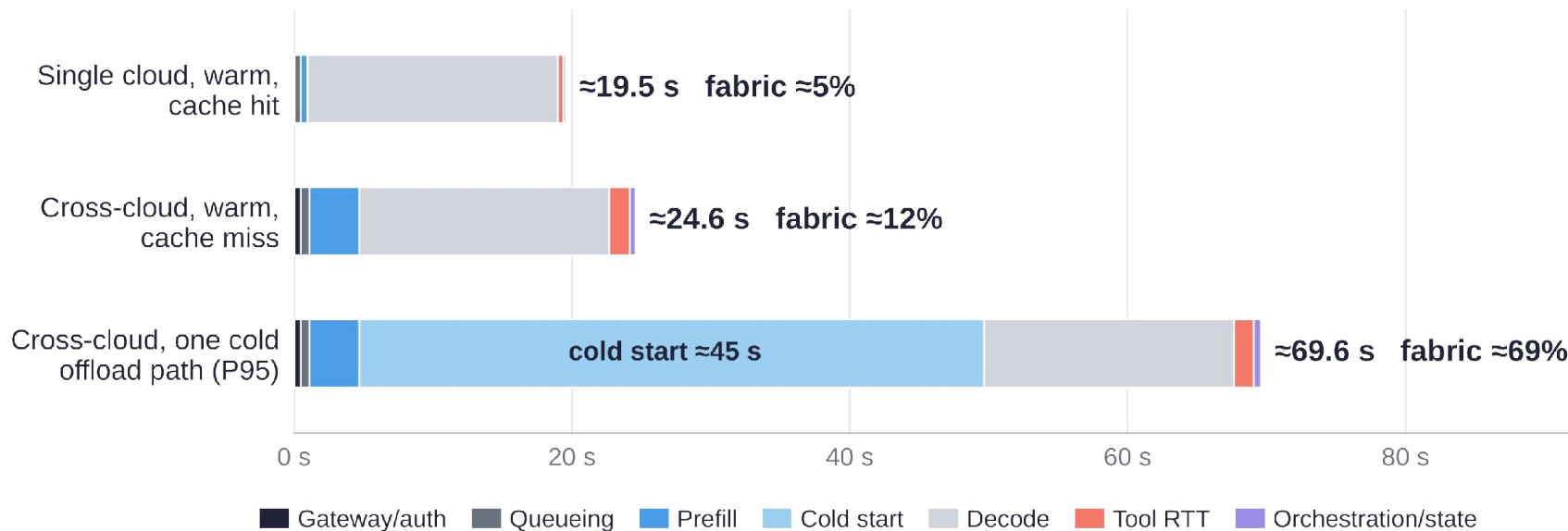
Post-pivot, must be idempotent, retried to completion

- 1987 theory → 2026 practice: SagaLLM (VLDB 2025) applies saga + compensation + validation to multi-agent LLM planning. LLMs alone "struggle with global constraint awareness"; the safeguards must be structural
- Encode the classification in the MCP tool definition (annotations), so the orchestrator can sequence compensatable steps before the pivot

Design rule: no tool gets exposed to an agent without a taxonomy label. Unlabeled = pessimistic default.

Richardson, Microservices Patterns (compensatable/pivot/retriable); Microsoft Azure Architecture Center, Saga pattern; SagaLLM, PVLDB 2025 (arXiv:2503.11951); MCP spec defaults.

End to end: decode stays flat, fabric sets the tail



Decode is the same 18 s in every bar. What grows is fabric, plus the prefill you redo when your cache doesn't cross providers. Tail latency is fabric latency.

Modeled from public anchors: MLPerf Inference v5.1 interactive limits (TTFT 0.45–0.5 s, TPOT 30–40 ms); Artificial Analysis provider TTFT and output speed; serverless-GPU cold-start literature (slide 11); inter-cloud RTT ~30–130 ms (Kentik Cloud Latency Map, CloudPing). Six model calls plus six tool calls. Fabric = everything except prefill and decode.

Per hop: fabric dominates time-to-first-token

Five components. Only one is the model.

$$T_{\text{TTFT}} = T_{\text{gw}} + T_{\text{q}} + T_{\text{r}} + T_{\text{pf}} + T_{\text{kv}}$$

Prefill is model-class. Gateway, queueing, routing and KV setup are fabric-class.

Calibrated to 4.5% mean error against four independent references from MLPerf, Splitwise, Mooncake and CloudPing.

Validated on four nodes — AWS us-east-1, GCP us-east4, RunPod Iceland, RunPod Illinois — for under \$10, reproducible from self-service accounts.

FABRIC SHARE OF TIME-TO-FIRST-TOKEN, PER HOP

	Same metro (measured)	Intra- continental	Trans- Atlantic	Trans- Pacific
Commodity path	72%	78%	84%	89%
100 Gb/s KV path	31%	57%	76%	86%

Same metro is measured; the rest is extrapolated with published RTT medians. Prefill stays flat with distance; every fabric term grows.



Operations beat geography

Utilization 0.5 → 0.8 adds ~3.4 s and enterprise-class auth adds ~3.0 s. Swinging cross-region RTT ±50% moves TTFT by at most 2.2%.



Agent fan-out breaks Poisson

Eight parallel tool calls makes traffic 8x burstier, and modeled queueing delay jumps 5.7x



Route on tails, not means

Fastest on average, slowest at the tail: P95 was 5.4x its median, vs ~1.3x elsewhere.: p95/median of 5.4 against 1.3–1.4 elsewhere.

The regime where TTFT behaves like a model metric is narrow: one metro, a fast private interconnect, warm auth and low utilization.

Three things to do Monday morning

1

Label every tool an agent can call: compensatable / pivot / retrievable + idempotent? destructive? Put it in the MCP annotations. Unlabeled = pessimistic default.

(Patterns 2 & 4)

2

Move workflow state out of your model providers. Pick a durable execution layer; decide worker placement as an SLO decision.

(Pattern 1)

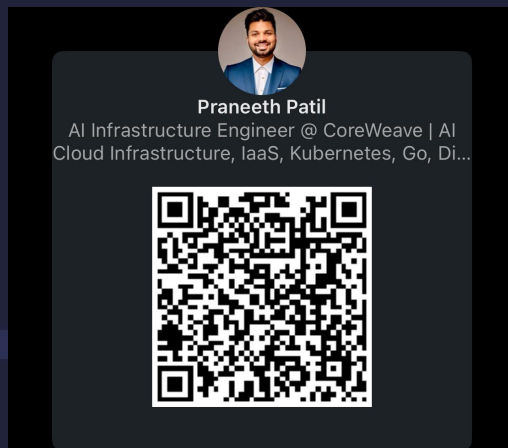
3

Write your latency budget in 7 lines. Measure one real workflow; find your decode-vs-fabric split; then decide caching, colocation, warmth, hedging.

(Pattern 3 + the number)

Agent reliability is not a model property. It's an architecture property.

Praneeth Kamalaksha Patil · CoreWeave



LinkedIn: [in/praneethpatil/](https://www.linkedin.com/in/praneethpatil/)

Backup: latency & cost model assumptions

Component	Anchor	Source
auth	10 ms same-cloud cached / 80 ms cross-cloud (TLS 2×RTT @ 40 ms)	TLS handshake math + inter-cloud RTT (Kentik/cloudping)
queueing	50–100 ms	MLPerf v5.1 server-vs-interactive scenarios
prefill	90 ms cache-hit / 600 ms 1.5k-token miss	MLPerf interactive TTFT limits; Anthropic ~85% TTFT reduction on cache hit
decode	15 ms/token × 200 tokens	Artificial Analysis provider TPS (66–480 TPS range)
tool RTT	4 ms same-cloud / 160 ms cross-continent round trips	Kentik Cloud Latency Map; cloudping.co
orchestration	20–40 ms/step durable write	Temporal-class event append, public docs
cold start	45 s (mid of 40–90 s)	serverless-GPU surveys; ServerlessLLM arXiv:2401.14351
Pricing	\$3/\$15 vs \$0.5/\$1.5 per M tokens; cache read 10%; H100 \$2.47–\$12.29/hr	provider price pages + 2026 surveys

References

- Flexera 2026 State of the Cloud
- IDC FutureScape 2026 press release, Oct 2025
- Gartner press release, Jun 25 2025
- AI Incident DB #1469 (PocketOS/Railway, Apr 2026) + #1152 (Replit, Jul 2025)
- Temporal Series D announcement, Feb 17 2026 + customer pages
- OpenAI Agents SDK + Temporal (Jul 2025)
- AWS APN blog, AgentCore + Temporal
- MCP spec tool annotations + MCP blog, Mar 2026
- CVE-2025-49596 / CVE-2025-6514 / CVE-2025-53109/53110
- Garcia-Molina & Salem, “Sagas,” SIGMOD 1987
- Richardson, Microservices Patterns
- Azure Architecture Center, Saga pattern
- SagaLLM, PVLDB 2025 (arXiv:2503.11951)
- MLPerf Inference v5.1 (MLCommons, Sep 2025)
- Artificial Analysis methodology
- ServerlessLLM (arXiv:2401.14351)
- vLLM cold start (arXiv:2606.07362)
- Kentik Cloud Latency Map; cloudping.co
- Anthropic prompt caching docs; OpenAI prompt caching docs
- AWS Bedrock cross-Region inference docs
- CloudZero / Spheron GPU pricing 2026
- Linux Foundation AAIF announcement, Dec 9 2025
- StackGen, State of Reliability 2026
- Patil, “TTFT is a Fabric Metric” (under review 2026)
- Splitwise (arXiv:2311.18677); Mooncake; CloudCast