PAPER · WORKSHOP TRACK

Decoupling Reasoning from Action: Architectural Impacts on Agentic Planning Consistency

Himaneesh Sompalle · Stonehill International School

Work done during internship at Emergence AI

OPENREVIEW.NET/FORUM?ID=YLFDFKM9DL

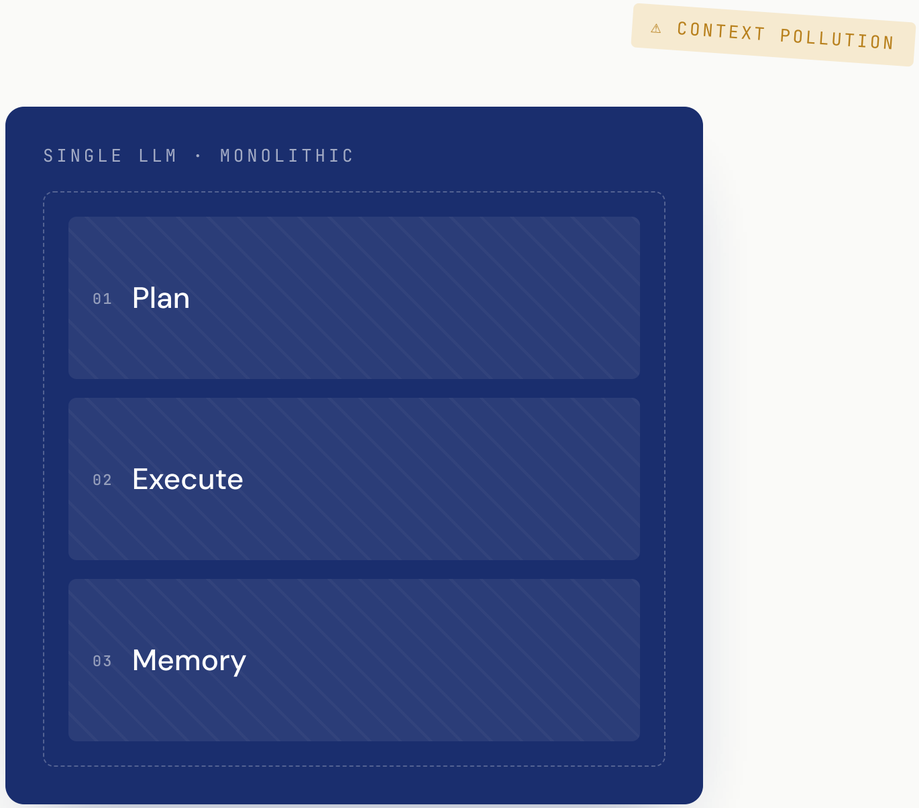
§ 1 · MOTIVATION

The Problem with Monolithic Agents.

Today's agentic systems collapse **planning**, **execution**, and **memory** into a single LLM call. As tasks grow, the context window fills with raw tool outputs, retries, and error logs.

The reasoning surface — the part doing the actual *thinking* — gets buried under operational noise.

> We measure this empirically across three architectural patterns and six ablations on `MCPBench` — the gap nobody had quantified.



§ 2 · THE THREE PATTERNS COMPARED

Three Architectural Patterns.

1

Monolithic

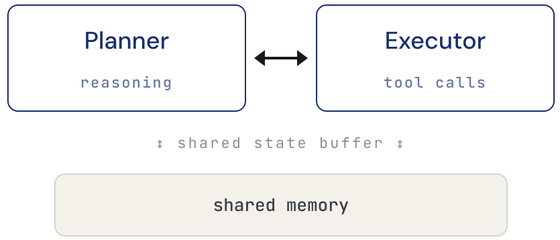
1-SERVER · SINGLE AGENT

One LLM owns everything. Maximum trace density — at the cost of context bloat as the task grows.

2

Partial Split

2-SERVER · PLANNER ⇄ EXECUTOR

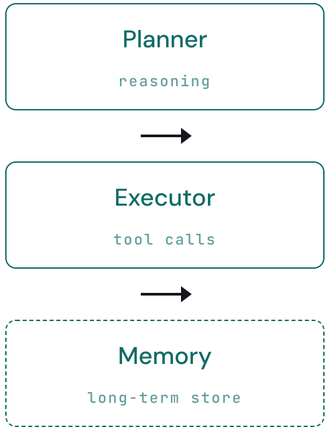


Roles split, but memory is shared. Reasoning is partially insulated from execution noise.

3

Fully Decoupled

3-SERVER · PLANNER → EXECUTOR → MEMORY



Each role is its own server. Memory is dashed — accessed only on demand. Highest insulation, lowest density.

§ 3 · METHODOLOGY

MCPBench · GPT-4o · 6 Ablation Studies.

18

BENCHMARK RUNS

3 architectures × 6 distractor levels

15

TASKS PER CONFIGURATION

drawn from the MCPBench pool

6–25

CONCURRENT DISTRACTORS

ratio swept ~1:1 → ~4:1

§ SIX ABLATION STUDIES · DISTRACTOR SWEEP

Noise floor →

A1	A2	A3	A4	A5	A6
Baseline	**Light**	**Moderate**	**Heavy**	**Saturated**	**Stress**
6 distractors	10 distractors	14 distractors	18 distractors	22 distractors	25 distractors

CONTROLS HELD CONSTANT

Temperature `T = 0` across all conditions

Identical system prompts, identical tool inventory

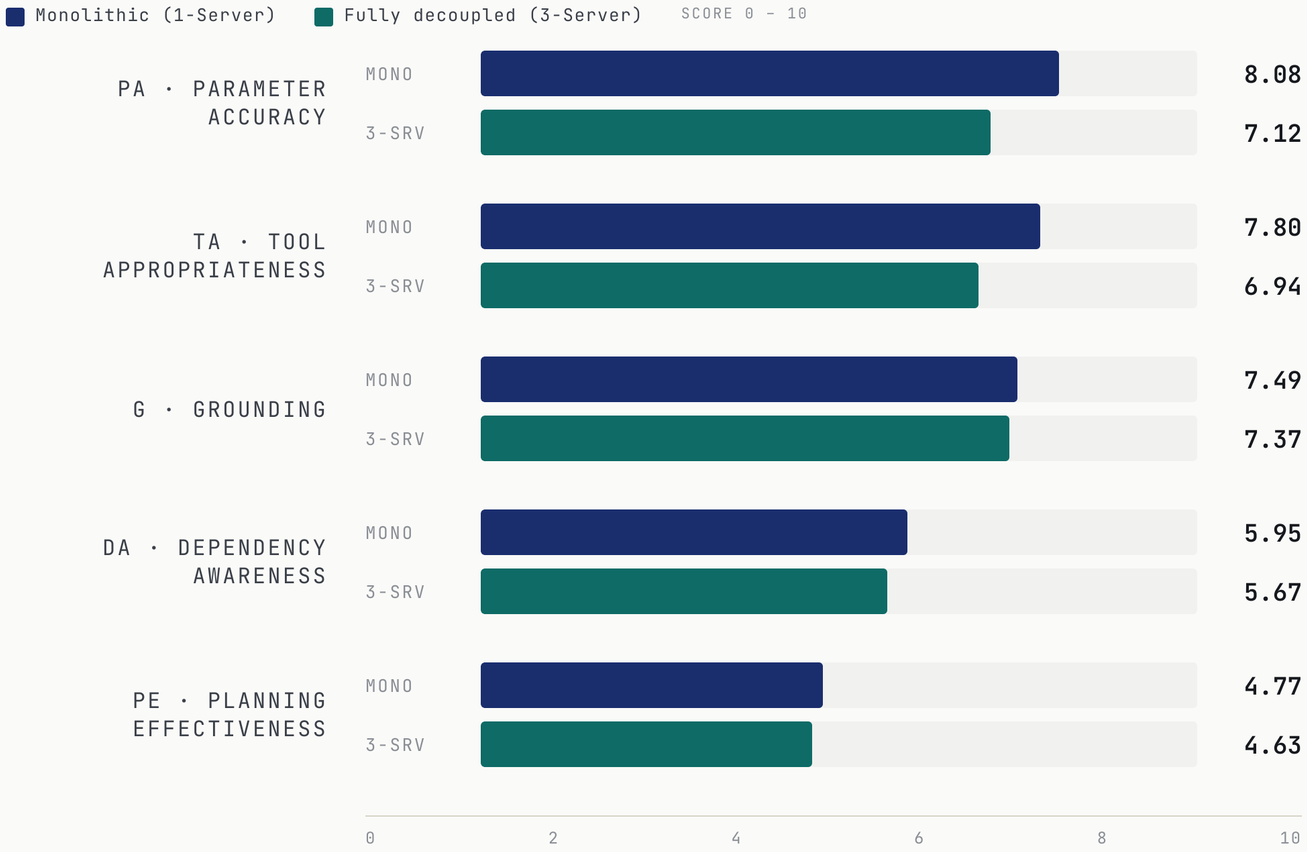
Same MCPBench task pool per architecture

FIVE QUALITY METRICS · SCORED 0–10

PE	DA	PA	TA	G
Planning Effectiveness	Dependency Awareness	Parameter Accuracy	Tool Appropriateness	Grounding

§ 4 · KEY FINDING · LOW-NOISE REGIME

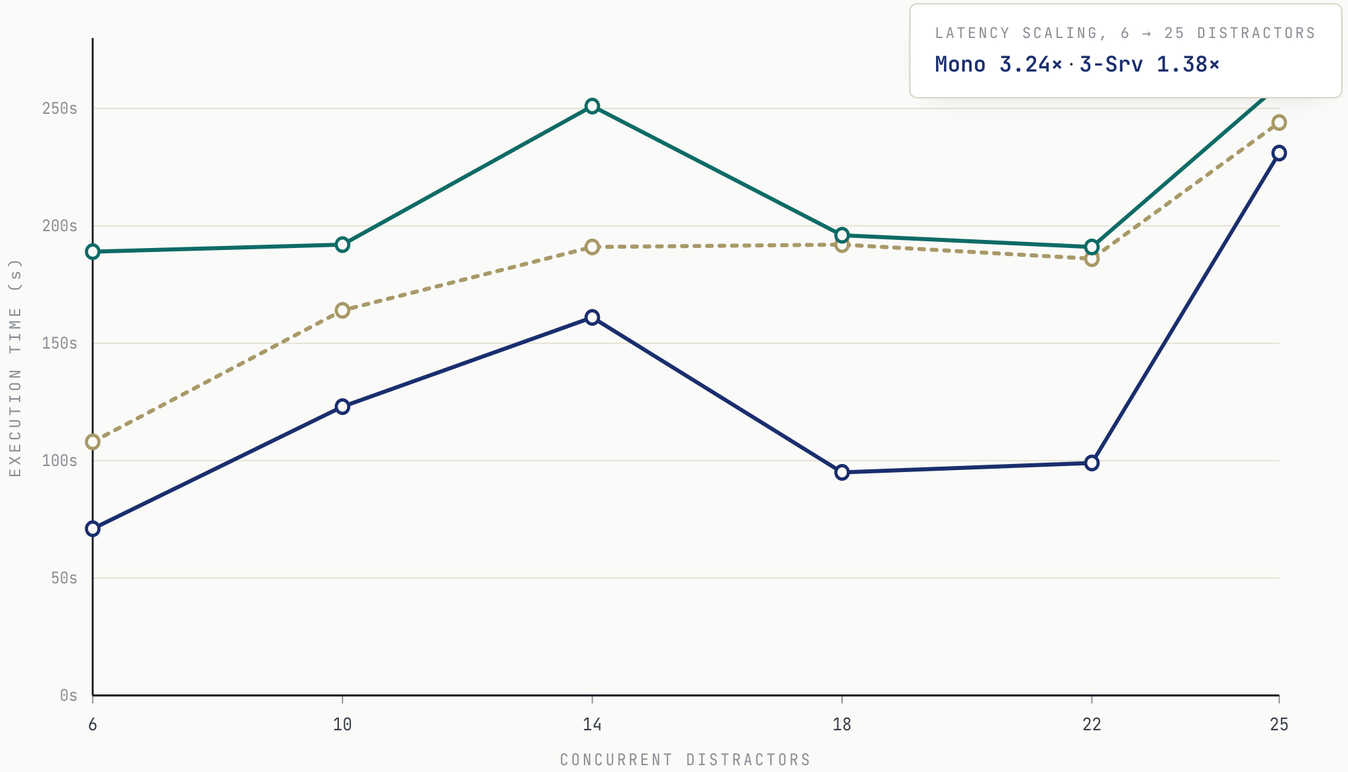
Monolithic Wins at Low Noise.



> **GPT-4o leverages raw execution traces as *implicit chain-of-thought* — collapsing roles makes reasoning denser, not noisier, when the noise floor is low.**
>
> SECTION 4.2 · INTERPRETATION

§ 5 · KEY FINDING · HIGH-NOISE REGIME

Role-Separated Wins at Scale.



Monolithic
71 → 231 s · 3.24×

Partial split
108 → 244 s · 2.26×

Fully decoupled
189 → 260 s · 1.38×

Monolithic explodes under load — 3-Server flatlines.

Diverging tails at 22–25 distractors expose the cost of context pollution: monolithic reasoning re-reads its own bloated trace on every step.

§ 6 · SYNTHESIS

The Context–Reasoning Trade-off.

REGIME A · LOW NOISE

Low Noise

→

Monolithic wins

Dense traces feed implicit chain-of-thought. Higher reasoning quality across PA, TA, G, DA, PE.

REGIME B · HIGH NOISE

High Noise

→

Role-separated wins

Insulated reasoning surface. Latency scales 1.38× instead of 3.24× as distractors compound.

ANTI-PATTERN · AVOID

! **Partial partitioning is the worst of both worlds.**

Splitting roles without splitting memory inherits the operational cost of separation and gains none of the reasoning benefits.

94.29%
TOOL SUCCESS

0
QUALITY WINS / 5

§ 7 · CONCLUSION

Takeaways.

1 Reliability is architecture-independent.

All three architectures hit 100% task success. The choice doesn't affect *whether* the agent finishes — only *how*.

2 Monolithic excels at micro-reasoning.

Dense access to its own execution trace lets it leverage raw history as implicit chain-of-thought.

3 Role-separated excels at macro-stability.

Under high distractor load, isolating the reasoning surface keeps latency and consistency in check.

4 Partial partitioning is an anti-pattern.

Splitting roles without splitting memory inherits the cost of both — and the wins of neither.

FUTURE WORK

Adaptive architectures · cross-model replication · PKG integration