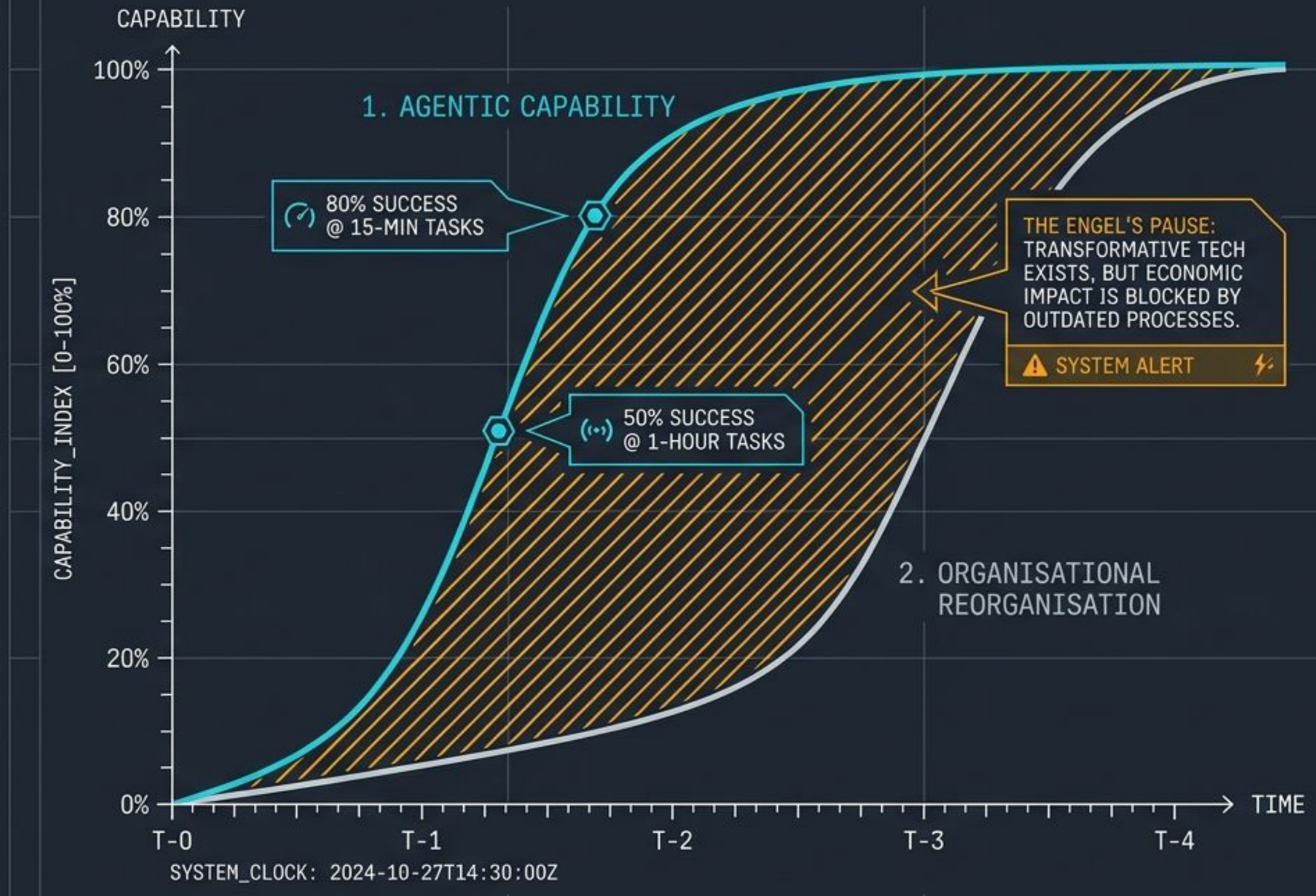
MIND_ROCKET_SERVICES // NEW_DEVX_VISION

Governing the Agentic Lifecycle

Moving Engineering Leadership from Human-in-the-Loop to Human-Above-the-Loop

V.02-2026 // SYSTEM: OPERATIONAL



THE BOTTLENECK IS NO LONGER TECHNOLOGY.

During the Industrial Revolution, steam engines existed for decades before factories reorganised to exploit them.

The 1980s productivity paradox saw massive IT investment yield zero output gain until processes changed.

Agentic AI is there now. The bottleneck is catching up on organisational muscle.

Dimension	Old DevX (Human-in-the-Loop)	New DevX (Human-Above-the-Loop)
Constraint	Human bandwidth (queueing behind people) 	Intent clarity and systematic review 
Primary Artefact	Stale documentation (Notion, PPT) 	Executable, versioned context 
Quality Assurance	Manual, line-by-line code review 	Governing and verifying the verification systems 
Goal	Fewer developers doing the same amount of work 	Unlimited code generation unlocking novel solutions 

ALERT: Accelerating code generation does not eliminate bottlenecks; it shifts them downstream to review and deployment.

SYSTEM_CLOCK: 2024-10-26T10:15:00Z

Three System-Level Decision Principles

ELEVATE THE NEXT CONSTRAINT

Fix code generation, the bottleneck moves to review. Fix review, it moves to deployment.

This is a continuous discipline gating prioritisation, discovery, and execution.

EVERYTHING AS CODE

If it is important, it is in Git. If it is in Git, it is tested on pull request.

The frontier is codifying the business problem itself (structured requirements, configs) so agents can write solutions at run-time.

HUMANS ABOVE THE LOOP

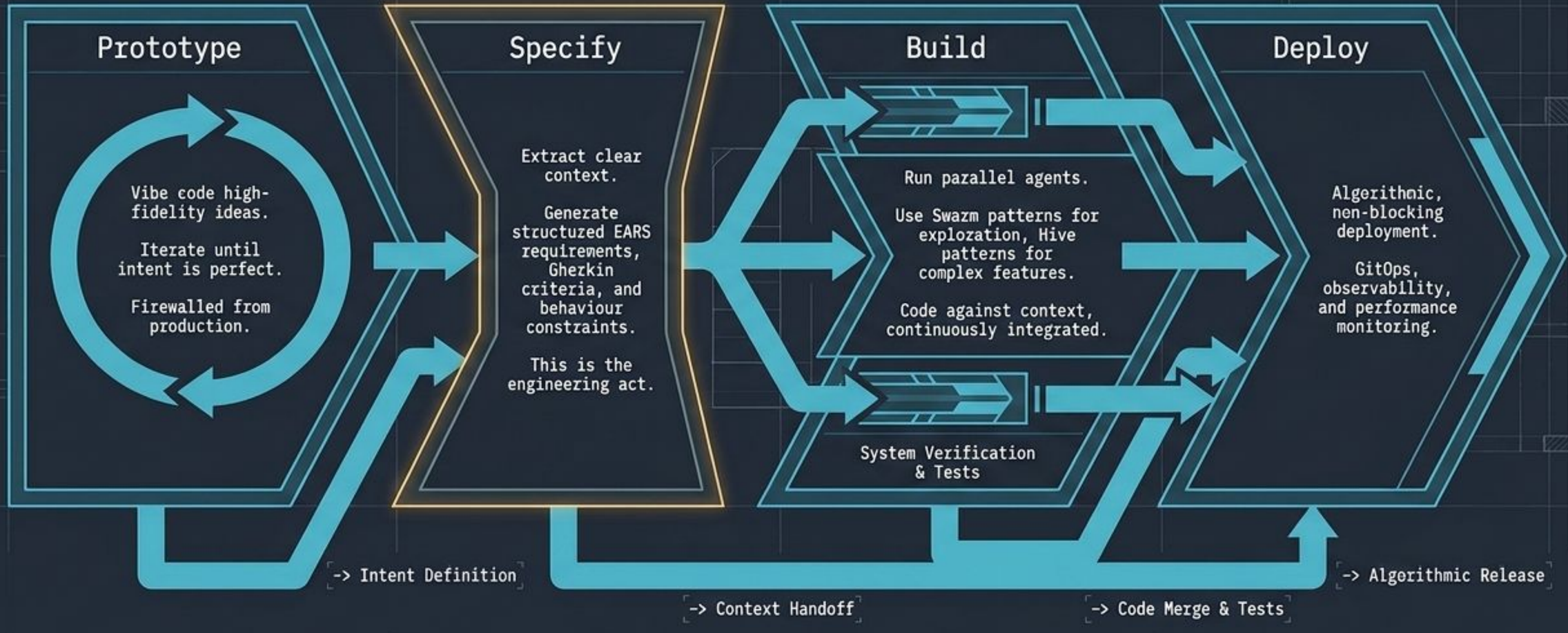
We do not hand off responsibility; we govern at a higher abstraction. Test the tests. Verify the verification systems. Classify work by risk.

SYSTEN_CLOCK: 2024-10-28T14:35:10Z

Speed and rigour are distinct modes, not competing choices.

	Prototype	Specify	Build	Deploy
Primary Actor	Human with AI Tools	Human Engineer	Parallel Agents	Algorithmic Pipelines
Mode	Creative-Dynamic (Vibe Coding)	Structured-Planning	High-Tech Execution	Robust-Operational
Output	Functional, janky prototype	Contracts, schemas, constraints	Production-grade code	Live systems
Risk Tolerance	High (Firewalled from production)	Zero (Strict engineering standards)	Zero (Tested continuously)	Zero (Algorithmic)

The Agentic Software Development Lifecycle (ADLC)



SYSTEM_CLOCK: 2024-10-28T18:15:602

The 5-Layer Agentic Toolchain

Operations Layer

CI Pipelines, CD & Cloud, Observability, Compliance.

`Function: Deploy & Govern.`

External Knowledge

Context Libraries, MCP Servers (Jira, Notion, Search).

`Function: Third-Party Resources.`

Intelligence

Workflows, Context Sharing, Model Routing, MCP Proxy.

`Function: Reasoning & Routing.`

Context

Model Memories, Task Specifications, Agent Rules, Project Context, Organisation Standards.

`Function: Knowledge & Constraints.`

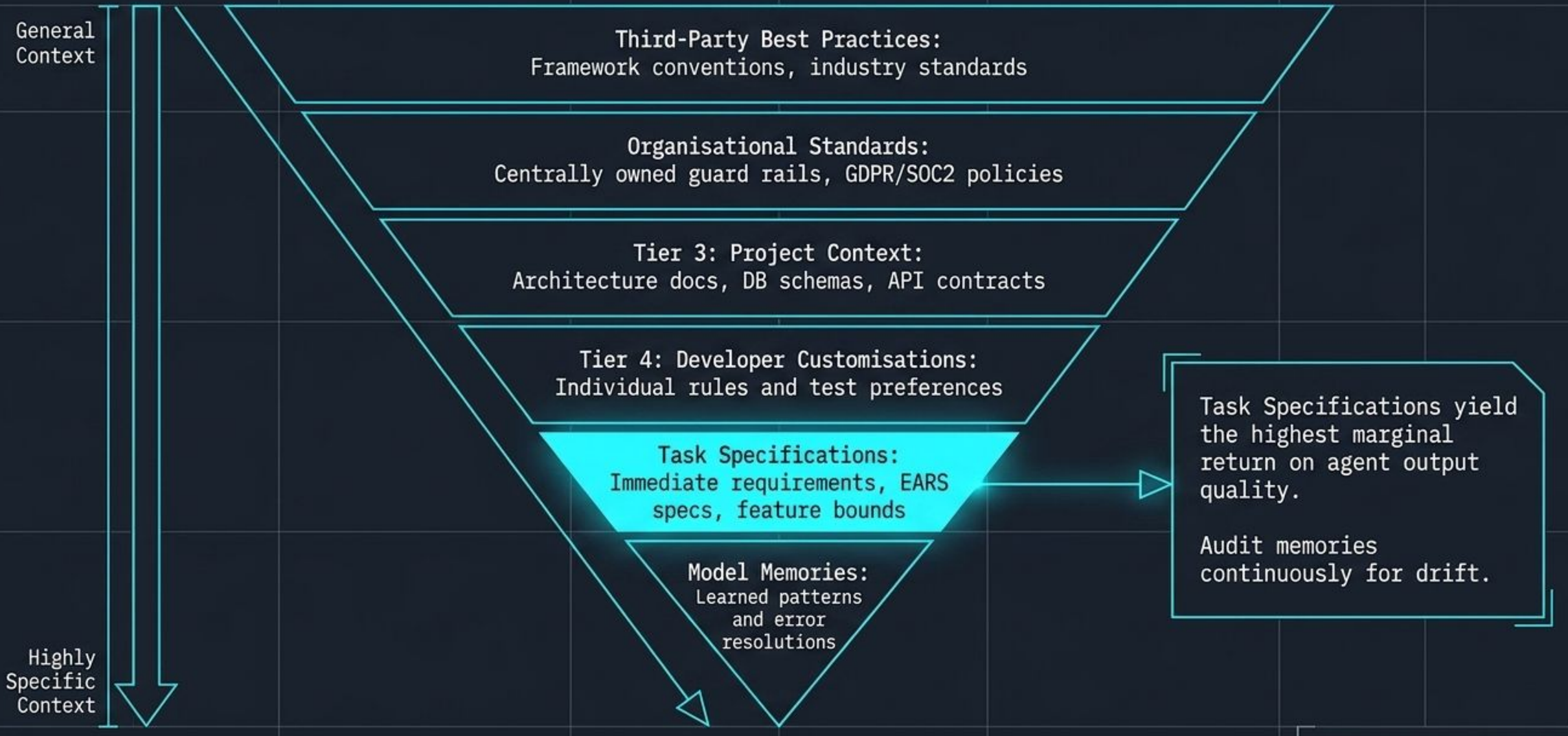
Execution Layer

IDEs, Isolated Containers, Cloud Agents, Repositories.

`Function: Run Code & Agents.`

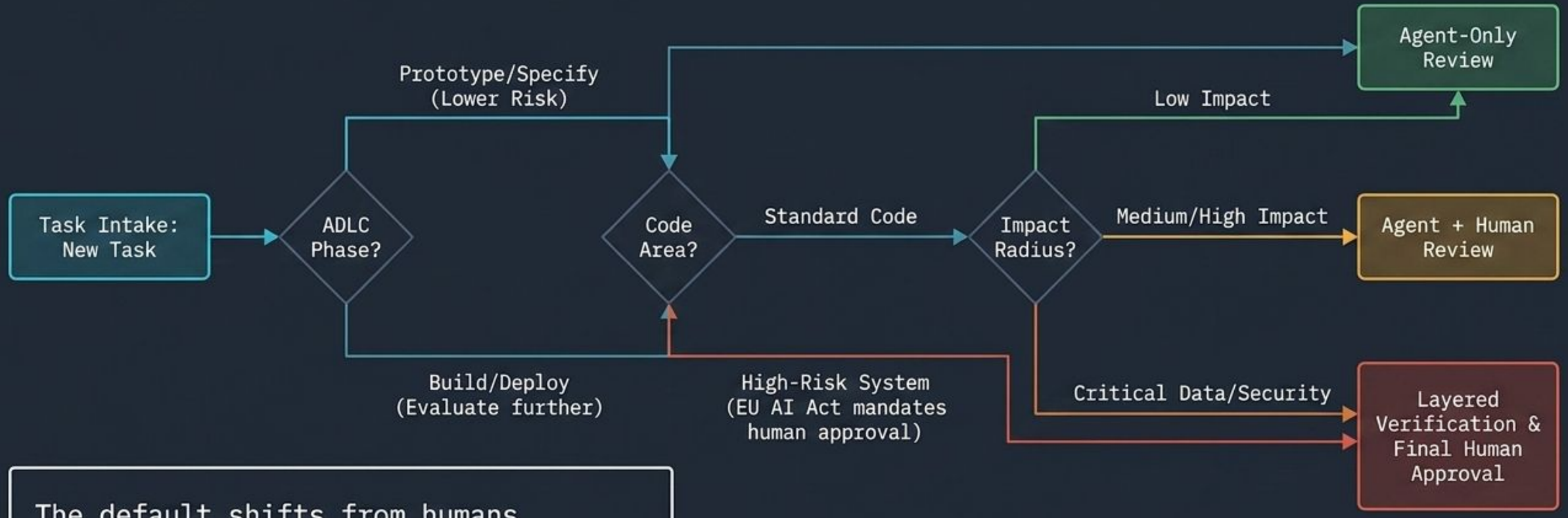
06:03:03

Agents require context, not instructions.



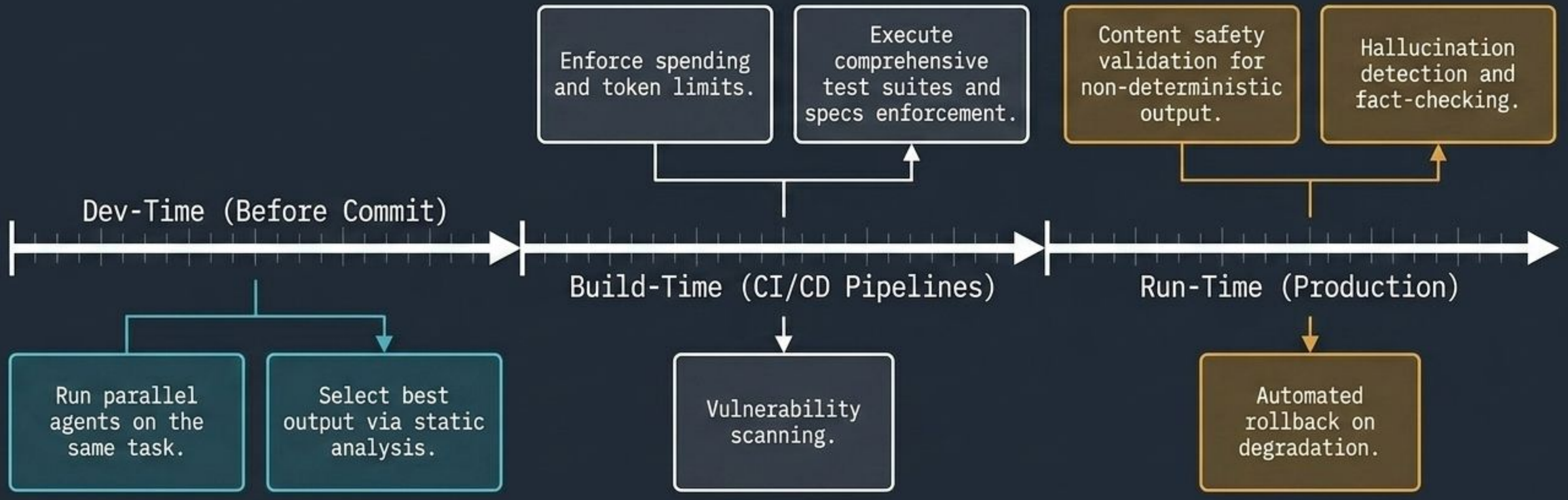
SYSTEM_CLOCK: 2024-10-28T18:15:60Z

Risk-Based Oversight: Systematic, not ad-hoc.



The default shifts from humans reviewing all agent work to agents reviewing agents, with humans involved strictly based on calculated risk.

Trust is engineered, not assumed.



Test to trust. Then trust the code.

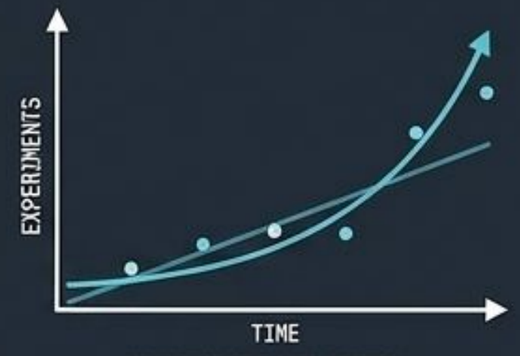
Navigate, Don’t Arrive: Measuring durable productivity.

Lag Indicators (What happened?)

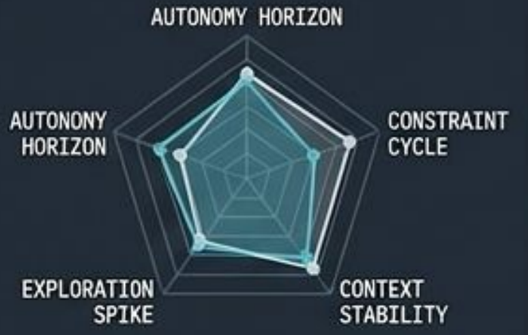
Cycle time (started to deployed)	Merges per developer per week	Bug resolution ratio.

⚠ **Warning:** A team can hit every lag target this quarter and still be six months from a capability gap. ⚠

Lead Indicators (What is coming?)



HYPOTHESIS VELOCITY

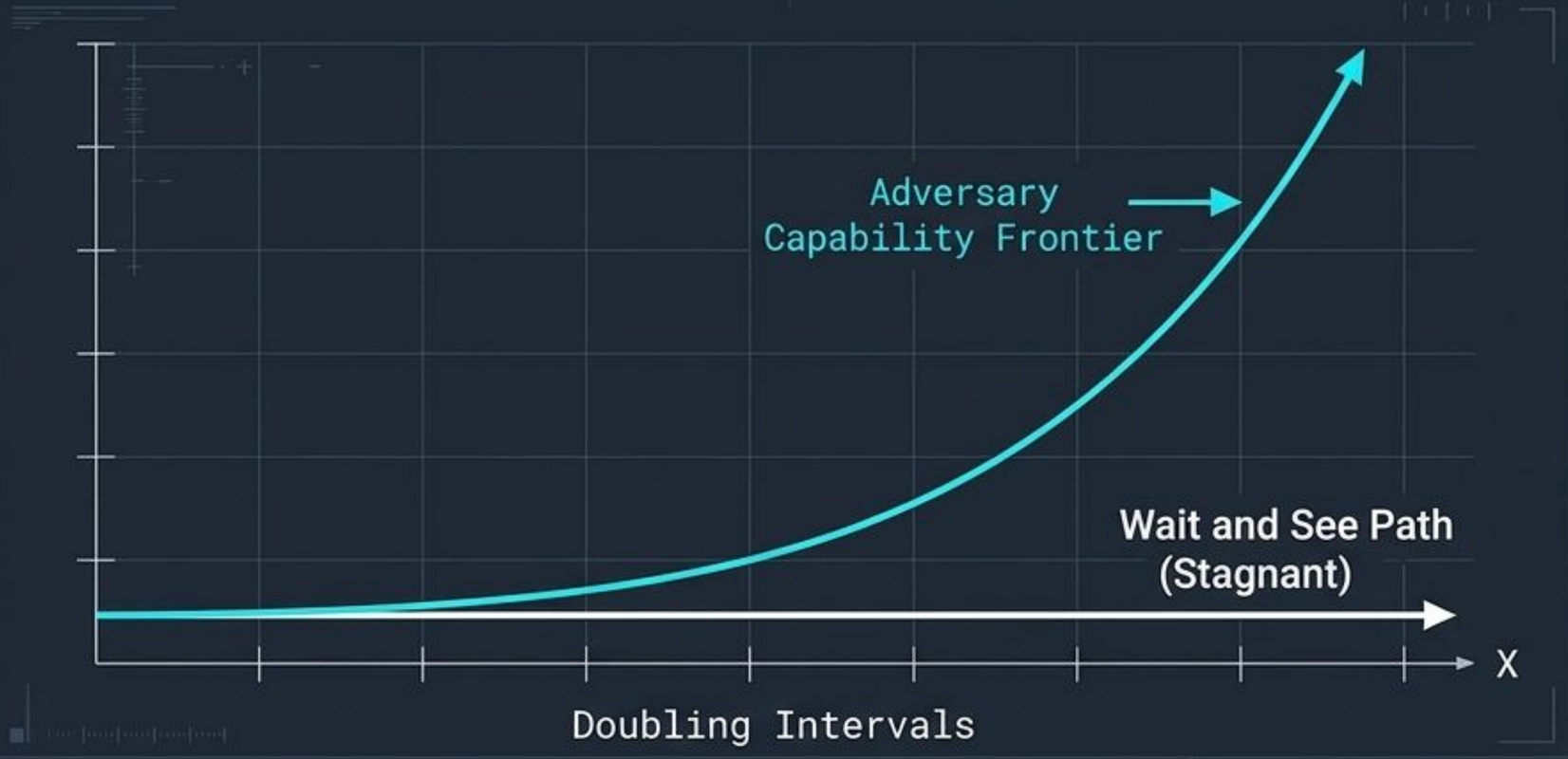


CAPABILITY EXPANSION

Hypothesis velocity (experiments per week)	Autonomy horizon trends (duration of human-free completion)	Constraint cycle time (speed of bottleneck removal).

⚠ **Warning:** If context files churn weekly and exploration spikes decrease, you are burning lead indicators to hit lag targets. ⚠

The compounding cost of delaying



The autonomy horizon doubles every 4 to 7 months.

Waiting a year does not mean starting a year behind. It means **starting against a capability frontier 2 to 4 doublings ahead.**

Late adopters do not get to skip the organisational learning; they do it **under extreme competitive pressure.**

Continuing to invest in human-in-the-loop processes calcifies the assumption. **The model itself becomes the constraint.**

The capability is here. The competitive window is open. Learn by doing now, or pay a premium later.