

A GUIDE FOR ENGINEERING LEADERS · OCTOBER 2026

The Guide to the Agentic SDLC

How software teams move from individual coding agents to one coordinated system of humans and agents across the whole lifecycle.



**Individual
productivity**

≠

**Team
productivity**

Coding agents made engineers faster. Teams only get faster when the whole lifecycle does.

INSIDE

The productivity plateau

AI as the System

Six stages of the agentic SDLC

Agentic loops

The platform stack

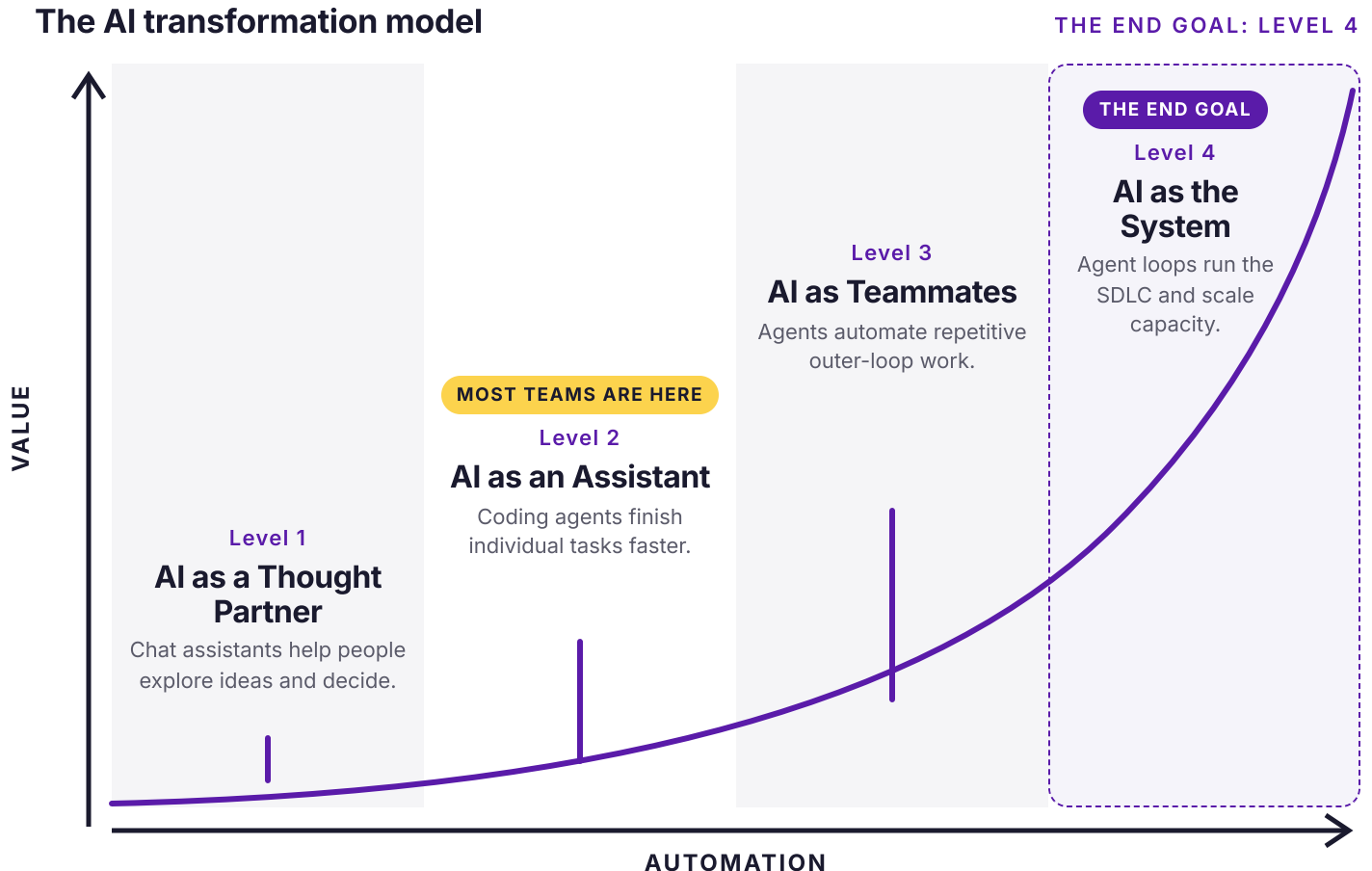
Security and governance

Getting started

Individual productivity is not team productivity.

Coding agents have made individual engineers faster, and more code is shipping. Yet most organizations do not feel faster. Code still moves through the same review queues, the same planning cycles, and the same release process. Individual adoption changes one engineer's workflow, not the team's delivery system, so the gains stay with the individual and nothing compounds.

The AI transformation model



For CTOs, the goal is Level 4. At Level 2 the gains stay with each engineer. At Level 4, humans and agents work as one system: loops run critical workflows from requirement to production, a shared Company Brain carries context between them, and capacity grows without adding headcount. Getting there takes a system where humans, agents, code, tools, and memory work together, not more licenses.

Why teams are hitting a productivity plateau

Engineering leaders bought coding agents expecting the whole company to ship faster. What they see instead is individual engineers writing more code while release dates, incident response, and roadmap speed barely move.

■ EXPECTATION

The company ships faster

■ REALITY

Only the engineer does

Individual adoption changes one engineer's workflow, not the team's delivery system. Code gets written faster, then waits in the same review queue, the same test cycle, and the same release process. The plateau ends only when the outer loop speeds up too.

WHERE THE PLATEAU SHOWS UP

[01]	PULL REQUESTS PILE UP FASTER THAN ANYONE CAN REVIEW THEM.
[02]	TESTING AND VERIFICATION LAG BEHIND THE CODE AGENTS WRITE.
[03]	RELEASES STILL WAIT ON MANUAL GATES AND HANDOFFS.
[04]	ON-CALL ENGINEERS SPEND THE NIGHT RECONSTRUCTING CONTEXT.
[05]	CUSTOMER FEEDBACK RARELY MAKES IT BACK INTO THE BACKLOG.
[06]	NOBODY CAN SAY WHAT THE AI SPEND IS PRODUCING.

CTOs are confused about what AI-native means.

Ask five engineering leaders what AI-native looks like and you get five answers. For one it means buying coding licenses. For another it means agents running production workflows. Without a common target, budgets, metrics, and roadmaps pull in different directions.

WHAT LEADERS MEAN BY IT	TRANSFORMATION LEVEL
// A LABEL FOR THE BOARD DECK, NOT A PLAN.	NOT DEFINED
// ENGINEERS BRAINSTORM WITH A CHAT WINDOW.	LEVEL 1
// EVERY DEVELOPER HAS AN AI CODING SEAT.	LEVEL 2
// SCRIPTS AND COPILOTS BOLTED ONTO EXISTING STEPS.	LEVEL 2
// AGENTS TAKE RECURRING TICKETS OFF THE TEAM.	LEVEL 3
// AGENT LOOPS OWN DELIVERY FROM IDEA TO PRODUCTION.	LEVEL 4

AI-native means AI as the system.

Humans steer and agents execute across the whole lifecycle, from requirements to production, on one shared Company Brain.

OUR DEFINITION

3x

Team velocity compared to 24 months ago

1 month = 1 quarter

Every month should feel like a quarter of progress, at minimum

Humans steer. Agents execute.

An agentic SDLC does not mean agents do everything. At every stage there is high-leverage work that belongs to people, such as direction, architecture, and quality ownership, and execution work that agents take on at speed. Coding is one stage of six. The other five make up the outer loop.

STAGE	HUMANS DECIDE	AGENTS EXECUTE	XHAWK AGENT
1 Requirements	Frame the problem and pick the direction	Synthesize customer feedback, stress-test options, draft specs	Product Manager, Brainstorming
2 Planning	Set priorities and accept the plan	Ground tickets in the codebase, surface dependencies and risk	CodeWiki, Chief of Staff
3 Implementation	Steer the approach and unblock edge cases	Turn tickets into PRs with repo context and conventions	Platform Agent
4 Review and test	Make the judgment calls and own the merge	Review every PR with deep research, verify against the source, simulate users	Verification Agent, PR Reviews, Synthetic Users
5 CI/CD	Own release gates and promotion	Put gates in place so only verified code ships to prod	Platform Agent
6 Operate and learn	Decide the fix and what comes next	Run runbooks, find root cause, analyze data, optimize spend continuously	SRE Agent, Data Analyst, Cost Optimization

Compounding Innovation

Agentic systems that get better over time share three properties.

[01] Context decides quality	An agent with your codebase, standards, and history produces reliable work. One without it does not.
[02] Verification outpaces generation	As agents write more code, review and testing have to keep up, or quality slips.
[03] Production feeds the backlog	Incidents and data findings return as tickets. That closed loop is where the system compounds.

Build the system one loop at a time.

A loop takes one recurring job, runs it from trigger to verified outcome, and repeats. Each loop is a team of agents with separate objectives. One does the work, another tries to find fault with it, and a person makes the call that matters. An agent grading its own work will pass it, for the same reason engineers don't approve their own PRs.

01 PR to merge

Trigger: a pull request opens

PR Reviews → Verification Agent → Platform Agent fixes → Human merge decision →

Company Brain

Usually the first loop to build. When agents write more code, PR volume rises and confidence becomes the constraint.

02 Ticket to PR

Trigger: a ticket is assigned

CodeWiki grounds the ticket → Human approves spec → Platform Agent implements →

Verification tests → Hands off to loop 01

The bottleneck is rarely code generation. It is specification, dependency discovery, testing, and coordination.

03 Alert to resolution

Trigger: an alert fires

SRE Agent investigates → Root cause posted → Human approves the fix → Fix PR or rollback →

Postmortem to Brain

On-call time goes mostly to reconstructing context. The agent arrives with it, and production calls stay with people.

04 Customer feedback to new features

Trigger: tickets, calls, and reviews arrive

Product Manager ranks themes → Data Analyst sizes impact → Human picks the bet →

Spec drafted, Synthetic Users test → Hands off to loop 02

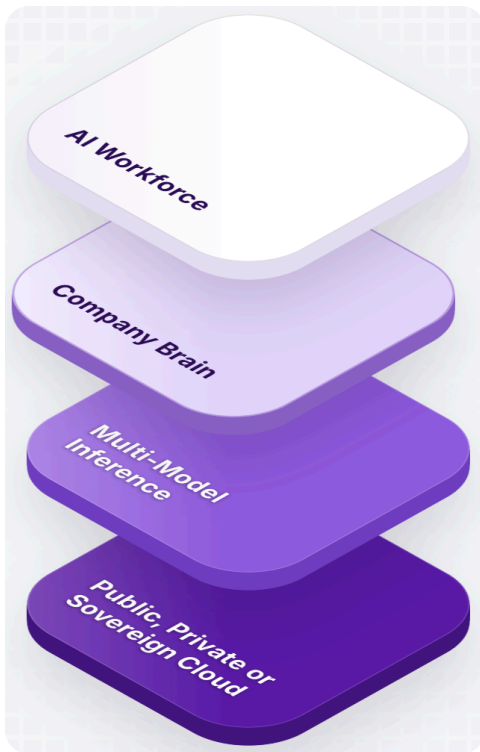
Closes the loop from production back to the backlog, so what customers say becomes the next thing the team builds.

AUTONOMOUS WORKFORCE

Design the loops: where people check in, how strong verification is, and whether tokens produce value. Measure outcomes per dollar.

Loops need a shared foundation.

Stitching point tools together means a separate login, data model, and context for every stage, and a custom integration project for every workflow. A single platform lets every loop build on, and contribute back to, the same foundation. XHawk keeps its layers separate and pluggable, so no layer depends on a single model or on one place where the agents run.



AI Workforce

Specialist agents for engineering, product, and analytics that call each other on long tasks.

Company Brain

A permission-aware index of code, docs, tickets, incidents, and conversations. What one loop learns, every loop reads.

Multi-Model Inference

Claude, GPT, Gemini, open-weight, and private models, routed per task. Bring your own keys.

Public, Private or Sovereign Cloud

Cloud, hybrid, or your own VPC. The same agents run in each.

What the foundation has to provide

Orchestration

Decides which agent runs next and passes context between stages.

Isolated sandboxes

Every run gets its own environment to build, test, and deploy.

Event triggers

Schedules, GitHub events, CI failures, Slack, and webhooks start loops.

Context access

100+ connectors, MCP servers, and an indexing API for internal systems.

Shared memory

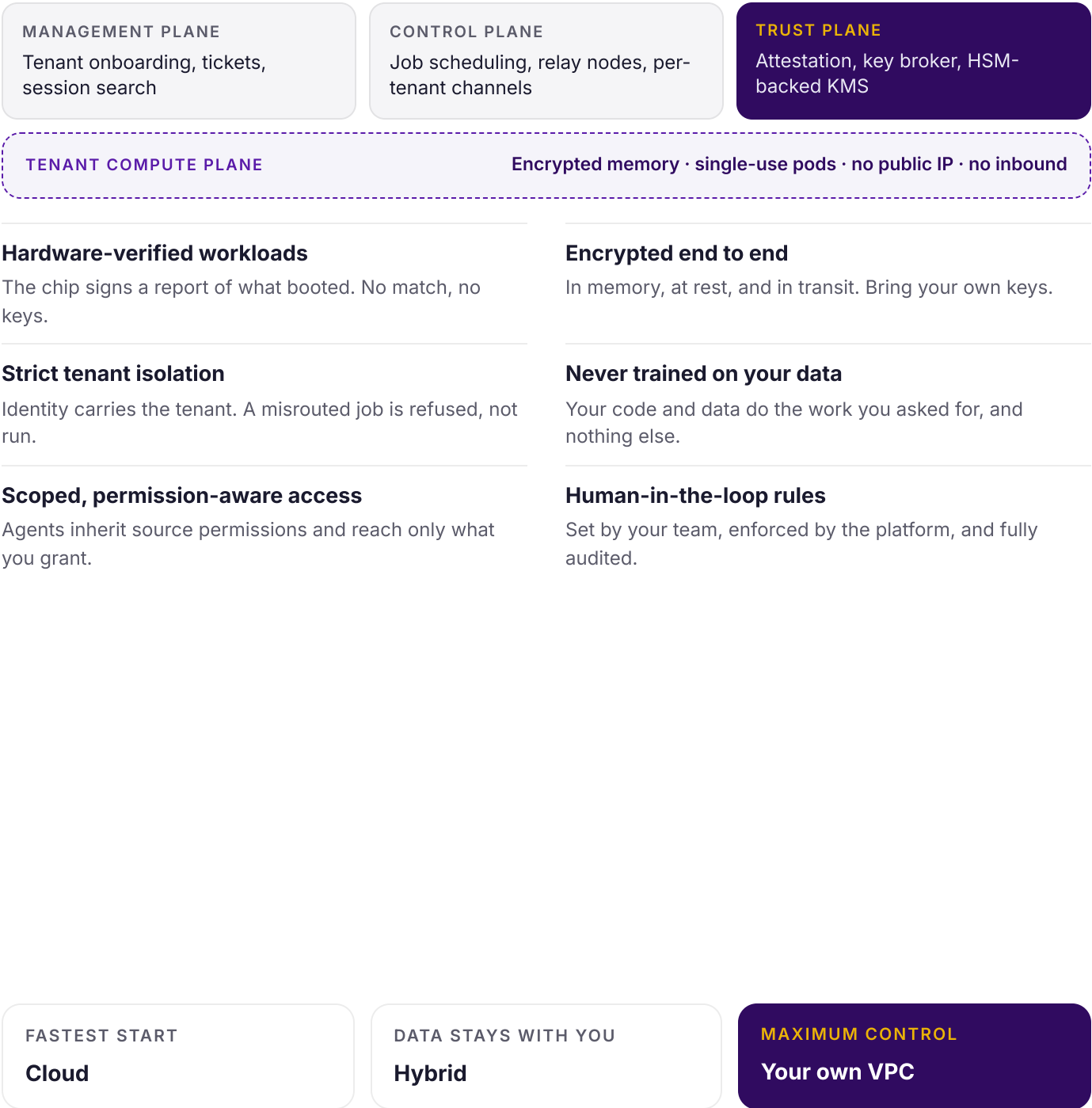
Corrections and incident patterns carry forward instead of resetting.

Cost visibility

Token and infrastructure spend tracked by agent, workflow, and team.

Governance is what lets you widen autonomy.

Leaders hand agents more of the lifecycle only when they can verify what the agents ran on, what they touched, and who approved it. XHawk runs every agent workload in a confidential computing enclave that proves what it is before it receives a key, secret, or job.



XHawk is in the SOC 2 compliance observation period.

Start where work waits longest.

The plateau breaks when you stop optimizing individual engineers and start optimizing the system they work in. Pick the stage where work queues up, build a loop around it, and add the next loop on the same foundation. Because every team reads from one Company Brain, each loop makes the others smarter.

ENGINEERING

Reviews, verification, CI/CD gates, docs, and incident response run around the clock.

PRODUCT

Customer feedback, specs, launch readiness, and synthetic user testing grounded in your data.

ANALYTICS

Plain-English answers from the warehouse, standard metrics, and continuous cost optimization.

From first connection to ROI in six weeks or less

- 01** **Connect your data, code, and business apps.** Agents get secure access in isolated sandboxes.
- 02** **Pick the first loop.** Start where work waits longest, often PR to merge.
- 03** **Configure agents and checkpoints.** Choose pre-built agents and decide where people approve.
- 04** **Trigger the work once.** Schedules, GitHub events, CI failures, Slack, or webhooks.
- 05** **Measure and add the next loop.** Track outcomes per dollar. Forward Deployed Engineers help with custom automation.

Model agnostic

Switch providers without rebuilding workflows.

Memory that accumulates

Every run leaves the next one better set up.

Location independent

The same agents in any cloud or VPC.



Build your agentic SDLC on XHawk.

Specialist agents for engineering, product, and analytics, on one Company Brain, in the cloud you choose.