ENGINEERING INTELLIGENCE ROADMAP

AI-Powered SDLC Automation

From AI-assisted engineers to fully autonomous agentic pipelines — a phased, production-grade roadmap.

Cursor | Claude Code | MCP | OpenShift | Self-Healing AI

THE VISION

From Human-Led to AI-Led Engineering

TODAY

- Engineer owns every action
- AI assists with autocomplete & review
- Human in the loop at every step
- Accountability stays with engineer

EVENTUALLY

- Agent owns the task end-to-end
- Human approves at key gates only
- Agent has full SDLC context
- Institutional memory outlasts any team member

The transition is gradual — each phase proves value, reduces risk, and builds the CTO conversation for the next.

PHASE 1

Cursor + MCP for Individual Engineers

Low risk, no new infra — prove value using existing Cursor licences

Jira

Read tickets, update status, create subtasks via mcp-atlassian

Confluence

Read specs & ADRs. Feed context into agent before coding starts

Bitbucket

Raise PRs, read commit history, inspect diffs

Jenkins

Trigger builds, tail logs — single API key from DevPlat, reusable

Grafana

Query metrics & logs read-only. Agent understands system state before changes

SonarQube

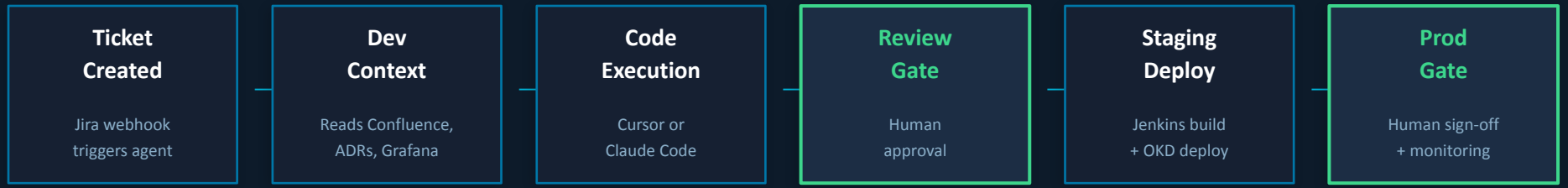
Code quality & security signals fed into review context

⚡ Write all rules and context as CLAUDE and CURSOR md-compatible markdown from day one — zero rewrite cost when we move to Claude Code later.

PHASE 2

SDLC State Machine — Our Orchestration Layer

We own this layer — not Cursor, not Anthropic. No vendor lock in.



= AI-driven transition

= Human approval gate

Portable by design — State machine logic is ours — plain code, our infra. Cursor/Claude Code is just the execution node. Swap it without rebuilding the orchestration.

Context passing between stages — Each transition carries structured context forward — ticket details, Grafana state, Confluence spec — so no stage starts blind.

Human gates are intentional — Code review and prod approval stay human-owned for now. This is what gets CTO buy-in. Automate them later once trust is earned.

PHASE 3

Self-Hosted Execution Engine

Swap the execution node to OKD — the entire pipeline now runs on our infra.

EXECUTION ENGINE OPTIONS

Claude Code (CLI)

RECOMMENDED

Terminal-native, 1M context window. Runs on OUR OKD as a pod. Usage-based billing, no seat lock-in.

OpenHands

OPEN SOURCE

Self-hostable autonomous coding agent. Full MIT license — survives any vendor acquisition.

LiteLLM + Custom Agent

FULL CONTROL

Our own orchestration on top of LiteLLM proxy. Maximum flexibility, maximum build cost.

PORTABILITY MATRIX

✓ WE OWN	SDLC State Machine
✓ WE OWN	MCP Server configs
✓ WE OWN	Rules & md files
✓ WE OWN	Agent execution loop (after Phase 3)
✗ VENDOR	Cursor Background Agent
✗ VENDOR	Cursor Skills marketplace
✗ VENDOR	Cursor cloud execution

PHASE 4

Self-Healing & Self-Training

The agent builds institutional memory — and improves itself with every task.

1

Session Memory

Current task context — ticket, code, Grafana state, conversation. Lives for the duration of the task.

2

Persistent Memory

Facts, decisions, preferences that survive across sessions. Who owns what, why module X is the way it is, deferred tech debt.

3

Procedural Memory

Reusable skill files written after every task. HOW to do things in this specific codebase. Refined each time a similar task runs.

THE SELF-HEALING LOOP

Task runs → Outcome analysed → Skill file written / updated → Similar future task benefits → Every 15 tasks: performance eval

🎙 Meeting transcripts → structured decisions extracted → indexed in persistent memory. Agent context eventually exceeds any individual engineer's.

PARALLEL UTILITIES

Human-Runnable Anytime, Independent of the Main Pipeline

Any engineer can invoke these on demand — no pipeline dependency, no approval needed.

Test Generation

- Point agent at any module or service
- Generates unit, integration, and edge case tests
- Reads existing test patterns in the codebase to stay consistent
- Raises a PR with coverage report attached

Codebase Onboarding

- New joiner asks natural language questions
- Agent answers using Confluence + code + commit history
- Explains 'why' not just 'what' — surfaces ADRs and decisions
- Gradually builds a living onboarding doc over time

CVE Vulnerability Sweep

- Pulls latest CVEs from NVD daily
- Cross-references your dependency tree
- Correlates with SonarQube findings for severity prioritisation
- Auto-raises prioritised Jira tickets with fix suggestions

ROADMAP AT A GLANCE

Phased Transition — Value at Every Step

01	**Cursor + MCP** Now • Individual engineers	MCP integrations for Atlassian, Jenkins, Grafana. Rules in md format from day one.	LOW
02	**State Machine** Next • Team-level	Own the orchestration. Cursor or Claude Code as swappable execution node. Human approval gates.	MEDIUM
03	**Self-Hosted** Later • Platform team	Claude Code or OpenHands on OKD. Full pipeline on our infra.	LOW-MEDIUM
04	**Self-Training** Goal • Org-level	Procedural memory, meeting context, skill accumulation. Institutional knowledge that never leaves.	MANAGED

The state machine is ours from Phase 2. Every phase after that is an upgrade to the execution node — not a rebuild.