Not a copilot. An autonomous engineering execution system.

EngineerX runs your engineering execution — the reviews, ECOs, requirements, documentation, approvals, releases, and handoffs — as one system. A farm of specialist AI agents does the work across every stage and holds quality at the gates you set, handing every judgment call to a human. Your engineers get back the **~70%** of their capacity that friction steals from building.

WHERE THE HOURS GO — ~70% TO FRICTION, ~30% TO BUILDING

~15%	~22%	~12%	~13%	~8%
Finding — rediscovering what the org already knows	Checking — reviewing, testing, chasing coverage	Creating — the PRDs, specs & docs nobody writes	Connecting — handoffs, impact & status across teams	Forecasting — risk you feel but can't see early

Friction estimates synthesized from the Stripe Developer Coefficient, GitHub Developer Survey, McKinsey Developer Velocity, and JetBrains Ecosystem studies — your stack and process determine actuals. The remaining ~30% is the building you actually hired for.

What makes it different

Reasoning, not scripting.
Give it an outcome — "get this PR merge-ready" — and it plans the path: reads the diff, drafts the test, suggests the fix, recovers from the failure. Work, not words.

Wired into your lifecycle.
Every answer comes from your repos, CI, tickets & history — grounded and cited, never guessed. Driven by your own patterns, ADRs & standards, not a generic linter ruleset.

Coordinated, not freelancing.
Bounded specialists — review, test, docs, risk — under one reasoning core, not one chat box. It drafts the fix, the test, the note and files it for review; nothing merges until a human says yes.

Across your execution

01 Ask the code anything
Repo Q&A · Project knowledge · Guided ramp-up · ask → req → code → test · Defect twins

02 Quality, built in — not bolted on
Test generation · Coverage from intent · Semantic review · Architectural intent · Real-RCA gates

03 The docs write themselves
PRDs & specs · Release notes & email · ADRs · Customer PDFs · Resolution notes

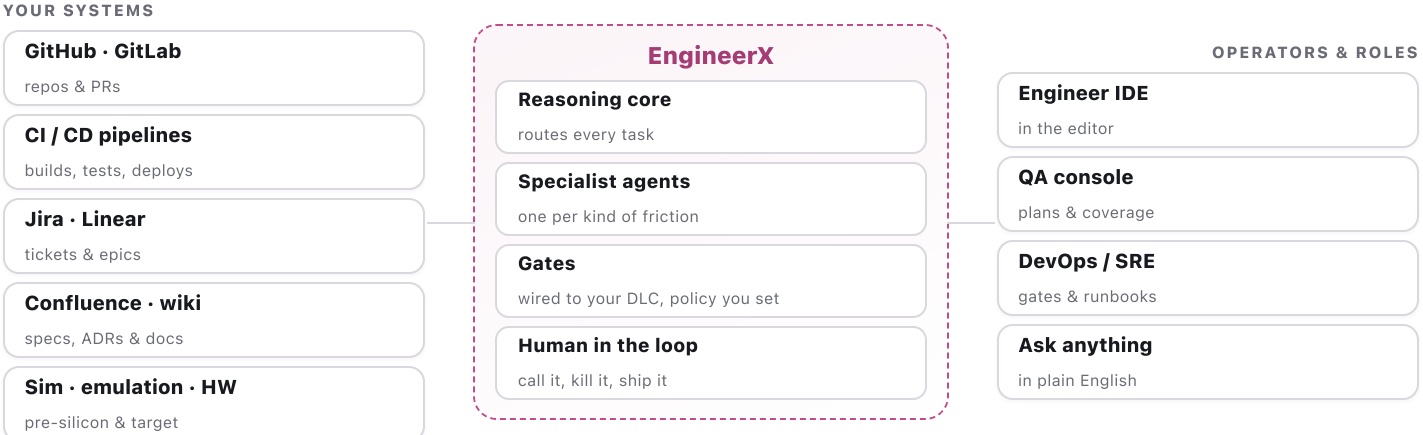
04 Handoffs that stop leaking
Impact maps · Conflict prediction · Runbook coverage · Auto standups · Action tracking

05 See the cycle before it costs you
Escape-risk scoring · Deployment risk & canary · Recurrence mining · Resource estimation · PR-time constraints

THE OPERATING MODEL

You own the gates. The agents run the execution.

Set the policy and the gates; a reasoning core reads your repos, CI, tickets and history, routes each task to the specialist that owns it, and drafts the work — the fix, the test, the note, the risk read — ready for a human to approve. Agents spin up on demand, do one job, and get out of the way. It runs inside your perimeter; your code never leaves.



Skip the slides. Run EngineerX on your own repos.

Book a demo: **30mins.com/studiox/call**
Or text directly: **+1 408 203 5641**

NO CODE · YOUR POLICY · YOUR GATES

You wire the farm to your lifecycle.

Choose which **agents** fire at each **gate** — pre-commit, commit, PR, merge, release — and what a **verdict** does: pass, fix, or escalate. Add a **specialist**, change a **policy**, stand up a new **check** — all in the console, no code and no waiting on a release. Discipline stops being voluntary and becomes the system's job.

THE GOVERNING PHILOSOPHY

It drafts. You decide. Discipline becomes the system's job.

A farm that touches your codebase can't be a black box — and the engineer stays in control of every call that matters. It grounds every answer in your own repos and cites the source, keeps a human on every judgment call, and runs inside your perimeter where your code never leaves — every verdict and action on an append-only trail.

THE ENGINEER STAYS IN CONTROL

1. **Grounded, always cited**
   Every answer comes from your repos, CI, tickets and history — with the exact lines, the ADR, the ticket. It won't invent an API or a fact.

2. **Human in the loop by design**
   It drafts the fix, the test, the note — then files it for review. Developers call what they want, kill what's wrong; nothing merges on an escalate until a person says yes.

3. **In your perimeter, fully logged**
   It runs inside your private cloud; your code never leaves. Every verdict and action lands on an append-only trail.

A DASHBOARD VS. THE FARM

LINTER / CI DASHBOARD	ENGINEERX
Flags the problem	✓ **Drafts the fix**
Same rules for everyone	✓ **Your patterns, your policy**
Runs when you remember	✓ **Fires at every gate, every time**
One check at a time	✓ **A farm across the whole DLC**
Passes or fails	✓ **Pass · fix · escalate — to a human**

Same engine as StudioX Core. The same AI Missions, Enterprise Knowledge layer, and 1,300+ integrations that run every StudioX deployment — the same pattern behind the resident lifecycle (ResidentX), network operations (NetworkX), the manufacturing line (FactoryX), and the financial close (FinanceX) — packaged for engineering. Wiring your repos and pipelines is a fast implementation, not a rebuild; the intelligence is already built.