Beyond Vibe Coding

How One AWS Partner Built an AI Engineering Platform on Kiro

Darren Mills, CTO • CleanSlate Technology Group • AWS Premier Tier Partner

About the Speaker

DARREN MILLS

Chief Technology Officer

CleanSlate Technology Group

Background

- CTO at CleanSlate Technology Group - AWS Premier Tier Partner
- 30 years as a consultant focused on application development
- 10+ years of AWS experience - 5x AWS Certified
- Instructional Associate at Georgia Tech in the Master's program
- AWS Ambassador - invite-only designation with limited global membership

Fun Facts

- Would rather climb mountains than the corporate ladder
- Amateur beekeeper, professional honey taster
- Yes, I hoard coins... no, you can't use them for the vending machine

The Productivity Paradox

Recent research on 100,000+ developers: AI writes far more code, but far less of it ships.

Cumulative effect of AI coding tools, by stage of the production chain:

+180%
Commits written

→

+50%
Projects

→

+30%
Releases shipped

Sync agents drove +741% lines of code and +65% pull requests — yet releases rose only ~20–30%.

The bottleneck moved

Writing code is no longer the constraint. Reviewing, integrating, securing, and governing it is.

Estimated AI–human elasticity of substitution: 0.25
— strong complementarity, not replacement. Gains attenuate at every human stage downstream.

And at the very end of the chain...

Across 4 app marketplaces:
more apps shipped since 2025,
but no increase in total usage.

Demirer, Musolff & Yang (MIT / Wharton), "Writing Code vs. Shipping Code," 2026. SSRN 6843118.

This is the case for governance.

Where We Started

We surveyed our engineering team. Ten engineers were using ten different AI tools.

10

engineers

10

different tools

The result

- Inconsistent delivery artifacts
- No governance controls
- No scalable operating model
- Every session started from scratch

Our approach

Most companies hand teams a tool and say “go use it.”

We did the opposite.

Define the operating model first — then embed the tooling into governed workflows.

Framework first. Tool second. The acceleration came from the governance framework, not the AI tool alone.

Traditional SDLC vs. AI-DLC

CleanSlate TECHNOLOGY GROUP

Why the methodology must evolve when AI becomes the primary executor.

Dimension	Traditional SDLC	AI-DLC (AI-Driven Development Lifecycle)
Lifecycle	Waterfall or Agile (sequential handoffs)	Continuous loop: Inception → Construction → Operations
Cycle Time	Weeks (2-week sprints)	Hours or days (“Bolts”)
Code Authorship	Human-written	AI-generated with human approval
Developer Role	Write code	Review, steer, and approve AI output
Governance	Periodic reviews and gates	Continuous automated gates + human approval
Security	Periodic scans (nightly, pre-release)	Embedded scanning at every phase
Context	Lives in meetings, wikis, human memory	Persists as structured artifacts in the repo
Parallelism	Limited by team capacity	DAG-based parallel execution of units

Why the methodology must evolve

- Writing code is no longer the bottleneck — review, security, and governance are. The methodology must match.
- Persistent context eliminates rework caused by decisions lost between sprints, meetings, and handoffs.
- Quality gates run continuously and mechanically — not as periodic checkpoints AI-speed delivery outruns.

AI-DLC in Practice: Every Discipline, Governed

CleanSlate TECHNOLOGY GROUP

Within AI-DLC's three phases, AI accelerates every engineering discipline.

INCEPTION

Project Management

Capacity planning, dependency mapping, and risk identification.

Requirements

Detects conflicts, resolves ambiguity, eliminates rework before code begins.

UI/UX Design

Generates accessible layouts from brand standards; responsive behavior.

CONSTRUCTION

Architecture & Design

Builds specs, validates against AWS Well-Architected, generates IaC.

Development

AI-assisted code with real-time governance and security scanning.

QA & Testing

Generates test cases from AC, builds automated suites, resilience testing.

OPERATIONS

DevOps & Infrastructure

Designs and implements IaC (CDK/Terraform), generates CI/CD pipelines, automates provisioning and monitoring. AI handles deployment orchestration; humans approve production changes.

Every discipline governed. Every output reviewed. AI executes — humans steer, approve, and validate.

Consistency → Quality → Speed

A mindset, not a scorecard. Each layer is earned before the next — speed is last.

Consistency

Repeatable, predictable output. The same standards on every engineer, every session.

Mechanism: steering files + the MCP shared-context server

Quality

Secure, reviewed, maintainable output validated before it advances.

Mechanism: hooks + spec gates + CI/CD quality gates

Speed

Only once the first two are proven do we scale velocity across the team and across roles.

Mechanism: skills + subagents + role expansion

Speed is the last layer, not the first. Speed without consistency and quality is just vibe coding with extra steps.

What Makes Kiro Special

CleanSlate TECHNOLOGY GROUP

An agentic IDE and CLI built by AWS — designed for how agents actually need to work.

IDE + CLI

VS Code-based IDE and a full terminal agent. Subagents, session persistence, runs alongside the AWS CLI.

Spec-Driven

Prompt → EARS requirements → design → tasks, with human gates. Specs become your sprint backlog.

Steering + Hooks

Markdown rules auto-load by file type. Hooks enforce governance in real time — block deploys, scan secrets.

Skills + Powers

Skills: on-demand expertise (SKILL.md). Powers: curated MCP + steering + hooks bundles. One-click install.

Custom Agents

Domain-specific personas with scoped permissions and tools — devops, security, cost, frontend, review.

Deep AWS Integration

AWS MCP servers are open source; Kiro bundles them into Powers with steering + hooks for turnkey workflows.

REAL HOOK — iac-safety-gate.kiro.hook (intercepts shell commands, forces a human confirm)

```
{ "when": { "type": "preToolUse", "toolTypes": ["shell"] }, "then": { "type": "askAgent", "prompt": "If this runs terraform apply / cdk deploy — STOP and confirm with the user first." } }
```

AWS-native: IAM Identity Center SSO • Amazon Bedrock • GovCloud (US) • customer-managed KMS • enterprise content not used to train models • IP indemnity

How the System Works: The Tiered Model

Kiro has a layered system. Each layer answers a different question.

AGENTS	**Who is doing the work?**	Domain-specific AI personas with scoped permissions
STEERING	**What rules does the AI follow?**	Markdown files that auto-load by file type and context
SKILLS	**What expertise does the AI have?**	On-demand packages loaded when the task matches
HOOKS	**What guardrails are enforced?**	Event-driven governance running in the background
SPECS	**How is work planned?**	Requirements → Design → Tasks (EARS format)

"It becomes an operational platform for how engineering teams deliver work, not just how they generate code."

Best Practices: Making the Primitives Work Together

CleanSlate TECHNOLOGY GROUP

Five building blocks, one capstone - how we actually run them across our repos.

Steering

Small, scoped files, versioned in Git. Auto-load by file type so context stays relevant.

IN OUR REPO

~17 files: 5 always-on + domain patterns that wake on matching code.

Skills

Codify a repeatable process once; it auto-loads by description match.

IN OUR REPO

8 skills: TOGAF assessment, 20-domain divestiture, IaC review, reporting.

Specs

EARS requirements, design, then tasks - with a human gate at each step.

IN OUR REPO

tasks.md is the backlog. Specs for real work; vibe mode for small fixes.

Hooks

Enforce, don't advise. Block risky actions; automate the critical path.

IN OUR REPO

9 hooks: iac-safety-gate, secret scanner, high-risk detection, drift.

Agents

One domain each, least-privilege tools. Humans approve every write.

IN OUR REPO

10 agents: devops (infra/ only), security & cost (read-only), review, api.

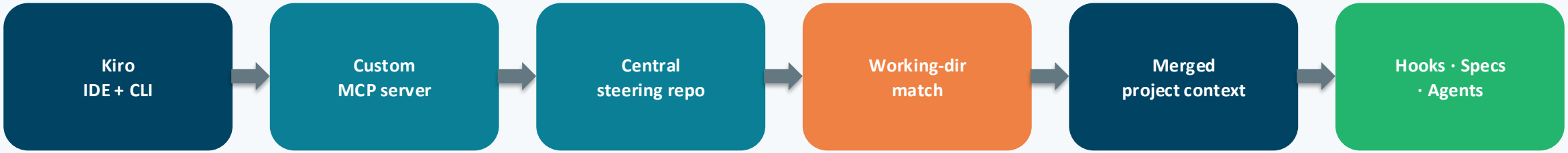
POWERS - package it once, hand it to the next team

A Power is Kiro's one-click bundle: steering + skills + hooks + (optionally) an MCP server. Our custom MCP server is the engine that distributes live context - wrap it in a Power and a new team inherits the whole governed setup.

Consistency (steering + skills) → Quality (specs + hooks) → Speed (agents)

Governance Through Shared Context

A custom MCP server - an open protocol for feeding AI external context - serves live steering across many repos and clients.



One server, many repos and clients — the right steering resolves automatically from wherever you're working.

Governance as code

- Rules version-controlled in Git, not a wiki nobody reads
- Update a standard once → every engineer gets it next session
- Project-specific overrides layer on top of shared standards

Onboarding that works

- New engineers run one setup script and get full project context
- Steering teaches the AI each project's conventions instantly
- The AI already knows the patterns — training feels natural

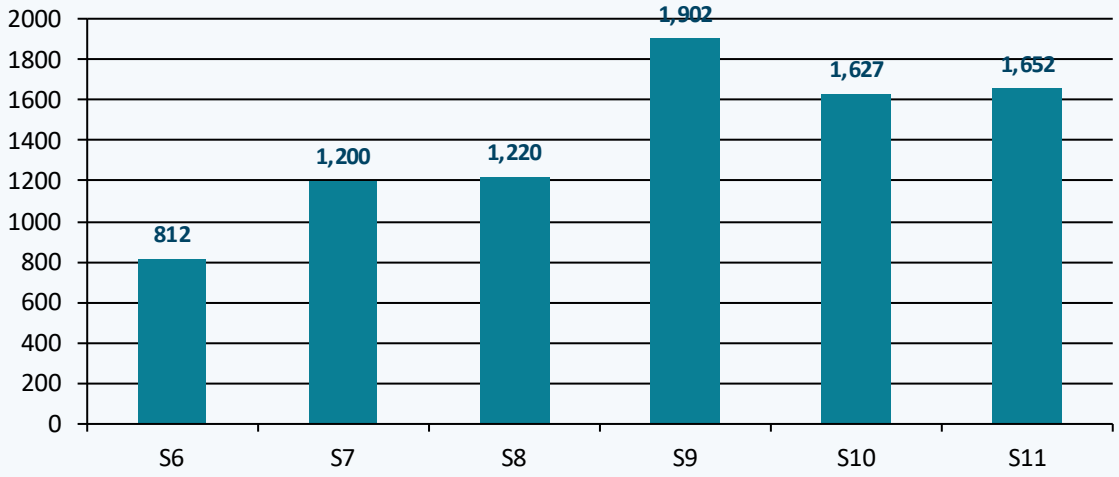
Think Group Policy for AI context: the right steering resolves from wherever you're working.

“The solution isn’t better prompting. It’s better context.”

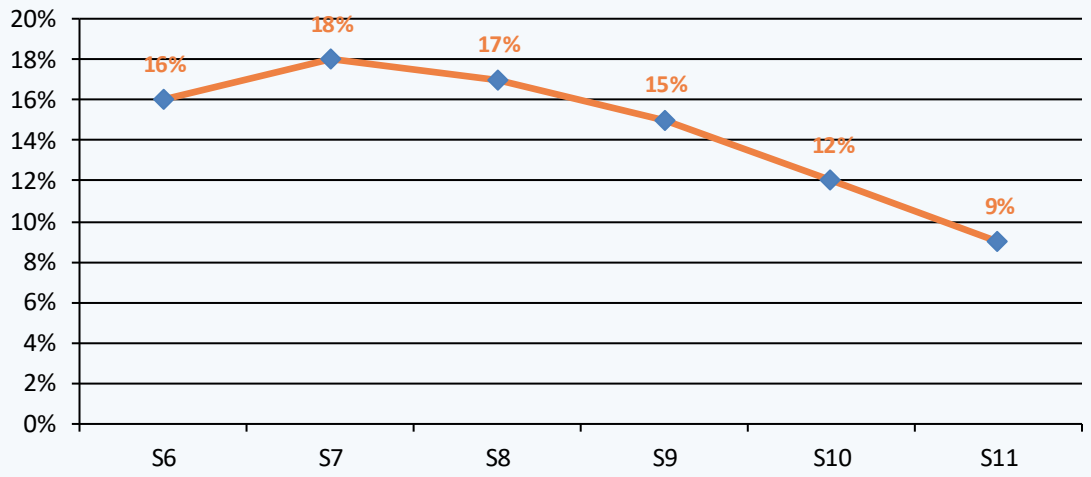
Our Velocity Story: Speed Exposed a Quality Bottleneck

We code ~2x faster now - but the gains only stuck once we raised requirements quality upstream.

Velocity — story points delivered



Quality — defect rate %



What we saw

- Coding got faster — but speed surfaced bottlenecks downstream, exactly what the research predicts.
- Our biggest unlock was raising requirements quality — it drove defects down while velocity stayed high.

What the data shows

- Velocity ~2x: 812 (S6) → 1,652 (S11), peaking at 1,902 (S9).
- Defect rate nearly halved: 18% (S7) → 9% (S11).
- Inflection at S9 — quality rose, defects fell, speed held.

Speed-first produced more defects. Quality-first produced durable speed — Consistency → Quality → Speed.

One Platform, Every Role

"If your team has a repeatable process, it can become a Kiro skill."

Engineers

CDK constructs, API development, IaC with governance hooks, the CLI running alongside the AWS CLI.

Architects

Architecture diagrams (Mermaid + AWS icons). Steering files encode and enforce design decisions.

Project Managers

Specs become sprint backlogs. Jira / ADO integration. Migration wave planning.

Sales

SOW templates with company branding; proposals generated in a consistent corporate voice.

Delivery

TOGAF assessments, IT divestiture planning, executive reporting, application modernization.

Every part of the business that adopted Kiro is benefiting in unique, non-development ways.

Branded Deliverables from a Prompt

Your brand system is just data. Codify it, and the AI produces on-brand deliverables.

Brand identity as code

Colors, logos, and icons codified in a skill. On-brand output, no manual formatting.

Consistent slide decks

Reusable PPTX layouts. Every deck matches the company template.

Professional PDFs

One-pagers, exec summaries, comparison docs with callouts and icon cards.

AWS architecture diagrams

Mermaid diagrams with official AWS icon packs. Describe it in English, get a diagram.

Example prompts

“Generate a one-pager comparing three DR strategies for the steering committee. Use the report-generation skill.”

“Generate a statement of work for a 12-week AWS migration. Use the SOW skill with our standard terms and pricing.”

This very deck was generated by that pipeline — branded PPTX, straight from a prompt.

Spotlight: M&A and Divestiture Delivery

On one engagement, we used Kiro to drive an entire corporate divestiture.

22 Architecture Diagrams

- Org, networking, identity, PKI, DR, CI/CD
- Steering enforces Mermaid + AWS icon styling
- CIDRs validated against source data, all in Git

Executive Documents

- Risk register, cost/TCO, migration timeline
- Go/no-go gates at each milestone
- Kiro flags conflicts between diagrams before ship

IT Assessment + Discovery

- it-org-assessment skill drives 20 domains
- Identity, DNS, backup, SIEM, patching, compliance
- Right questions in order + gap-analysis templates

Branded Deliverables

- report-generation skill emits PPTX + PDF
- Strategy decks, exec check-ins, deep-dives
- Company branding applied automatically

An entire divestiture — discovery, architecture, planning, and executive reporting — driven through one governed platform.

AI-Driven Methodology as a Skill

We codified consulting methodology as Kiro skills — the AI drives the process, not just answers.

Enterprise Assessment

TOGAF ADM phases. Score apps on technical health x business value.

8Rs Rationalization

A decision tree per application, across all 8Rs: Retire through Reinvent.

IT Capability Rationalization

Rationalized capabilities across 100 applications to identify consolidation and standardization.

8Rs decision tree (codified in the skill)

Still needed? NO → RETIRE
Not ready? → RETAIN
SaaS meets 80%+? → REPURCHASE
Hypervisor move? → RELOCATE
Minimal changes? → REHOST
Managed services? → REPLATFORM
Cloud-native needed? → REFACTOR
Transform the business? → REINVENT

Real skill descriptor — Kiro auto-loads it on description match:

```
skill: enterprise-assessment
Purpose: TOGAF application portfolio analysis,
  rationalization & cloud migration planning
```

Lessons Learned

CleanSlate TECHNOLOGY GROUP

1 Adoption matters more than the tool.

Unstructured adoption breeds inconsistency and governance gaps. Value comes from shared practices, not the tool itself.

2 Build the framework first.

Define steering, standards, and output expectations before anyone opens the tool. The foundation makes everything downstream reliable.

3 The limit is how you think about using it.

Architecture, reports, plans, SOWs, full workflows — all possible with intentional guidance. The ceiling rises as your team gets better at directing the AI.

4 Meet engineers where they work.

The CLI gives the full agent next to the AWS CLI. Skip spec overhead for small fixes — use vibe mode. Know the sweet spot.

Consistency, then quality, then speed. Speed without the first two is just vibe coding with extra steps.

Getting Started

If you want to try this — start small, let steering guide the AI, then expand.

1 Install + Configure

- Install Kiro from kiro.dev
- Free tier to start
- Create essentials.md + agents.md

2 Start Small

- Docs, bug fixes, “how do I” questions
- Let steering files guide the AI
- Build trust in the workflow

3 Expand

- Try a spec: requirements → design → tasks
- Add skills, hooks, custom agents
- Build your own MCP server

Resources

kiro.dev — IDE + CLI download
kiro.dev/docs/specs — spec-driven development
kiro.dev/docs/steering — steering files
awslabs.github.io/mcp — AWS MCP servers

Worth reading:

- From Spec to Production: a 3-week drug-discovery agent
- Accelerating GovTech development with Kiro
- Automating SOC 2 compliance with Kiro

CleanSlate TECHNOLOGY GROUP

Thank You

Questions? Happy to dig into any of this.

Darren Mills

Chief Technology Officer • CleanSlate Technology Group

darren.mills@cleanslatetg.com

cleanslatetg.com • (866) 885-3249

aws PARTNER Premier Tier Services

BEST PLACES TO WORK IN INDIANA 2023 2024 2025

Join our Office Hours

Thursday, June 25 • 3:30–5:00 PM ET • Virtual (Zoom)

Bring your questions on AI, architecture, data, and AWS modernization — an open discussion with our speakers and technical team.

Scan to continue the conversation

Slides, resources, speaker contacts, and the Office Hours invite.