Xiaotime *Labs*

A position paper · Category argument

GRC*Dev*SecOps

Why governance belongs in the pipeline, not the report.

DevOps closed the dev-ops wall. DevSecOps closed the dev-sec wall. **GRCDevSecOps is the next wall.** Governance, risk, and compliance still run weeks behind engineering. They do not have to. *This paper is about what changes when they stop.*

Position paper.
For CISOs, heads of platform, and GRC leaders who have noticed that compliance automation is still running weeks behind engineering.

Version
1.0
April
2026

New engineering disciplines get new names when the old names stop describing how the work actually gets done. DevOps. DevSecOps. Now this.

The lineage

DevOps was a rename of what happened when operations stopped being an adjacent team and started being part of how software shipped. The wall between development and operations came down because the wall was slowing everything.

DevSecOps was the next wall. Security stopped being a pre-launch review and started being part of the pull request. Threat modeling moved left. Dependency scanning moved into CI. Static analysis ran on every commit. The old model of security-as-gate was incompatible with the cadence of deployment.

GRCDevSecOps is the next wall. Governance, risk, and compliance are still organized around the annual attestation model. A SOC 2 letter, refreshed once a year. A heat-map dashboard, refreshed once a quarter. An auditor visit scheduled six months out. Meanwhile, the thing being governed ships every day. The wall between engineering and GRC is the one DevOps and DevSecOps left standing.

2008 ISH

DevOps

Dev ↔ Ops wall.
Closed by CI, CD,
infrastructure as code.

2012 ISH

DevSecOps

Dev ↔ Sec wall.
Closed by SAST, SCA,
security as code, shift-left.

NOW

GRC*Dev*SecOps

Dev ↔ GRC wall.
Closed by policy-as-code,
continuous evidence,
pipeline-native remediation.

The gap

The friction shows up in a familiar pattern. Compliance finds gaps late in the cycle and creates rework. Evidence goes stale the moment it is collected. Developers resent audit as a blocker. Auditors get snapshots instead of continuous assurance. A security review takes months. A deal closes or it does not. The cadence mismatch is the visible symptom. The cost is the rework the mismatch creates. This is not a process problem. It is an architecture problem.

Compliance automation platforms were built for a slower world. They pull evidence on a schedule from a catalog of integrations. They report on the state of controls to auditors. That model worked when the state of controls changed slowly. When the thing being governed deploys 40 times a day and adds a new dependency every hour, polling-based reporting is always stale, and every stale data point is potential rework.

AI is widening the gap fast. The EU AI Act, ISO/IEC 42001, NIST AI RMF, and the Cloud Security Alliance's AI Control Matrix all land in the next 18 months on organizations with AI systems already in production. None of them map cleanly onto integration-poll reporting, because AI behavior is not something you poll. Prompts. Model versions. Retrieval corpora. Agent tool calls. Output distributions. Guardrail events. These are governance signals the reporting layer was never designed to observe.

The gap is not going to close on its own. It will close when the way GRC operates changes shape.

The principle

GRCDevSecOps is governance at the pipeline layer.

Policy runs where the code is written. Evidence is emitted as a side effect of shipping. Remediation routes through the same gates the code does. The reporting layer still exists. Dashboards, audit packages, board views, auditor evidence exports. But the reporting layer now reads from a continuously-produced event stream instead of a quarterly pull.

If DevSecOps moved security into the pull request, GRCDevSecOps moves governance into the pull request. Same cadence. Same gates. Same signed audit trail.

Four principles of GRCDevSecOps

These are architectural principles, not tool recommendations. They hold across any specific implementation.

1

Policy-as-code at the pull request

Governance rules run where the change is happening. Not in a weekly spreadsheet. Not in a dashboard someone looks at once a month. In the pull request, before the merge, visible to the person or system making the change. Policy that can run in CI is policy that can enforce. Policy that only lives in a document is policy that gets argued about.

2

Evidence as a byproduct of building

Signals the system already emits are governance signals. Version control events. CI results. Dependency scans. Cloud configuration. Identity telemetry. AI operational behavior. Subscribe to them and project them into a control register that is always current. Evidence stops being a collection project and becomes a consequence of the system running.

3

Autonomous remediation on the same rails

When a control fails or a finding lands, an agent proposes the fix and opens the pull request. Inspection runs against policy. A human approves at merge. The change ships and is signed. Detection, correlation, and proposal happen autonomously. Approval and merge stay human. Build and remediate run on the same rails.

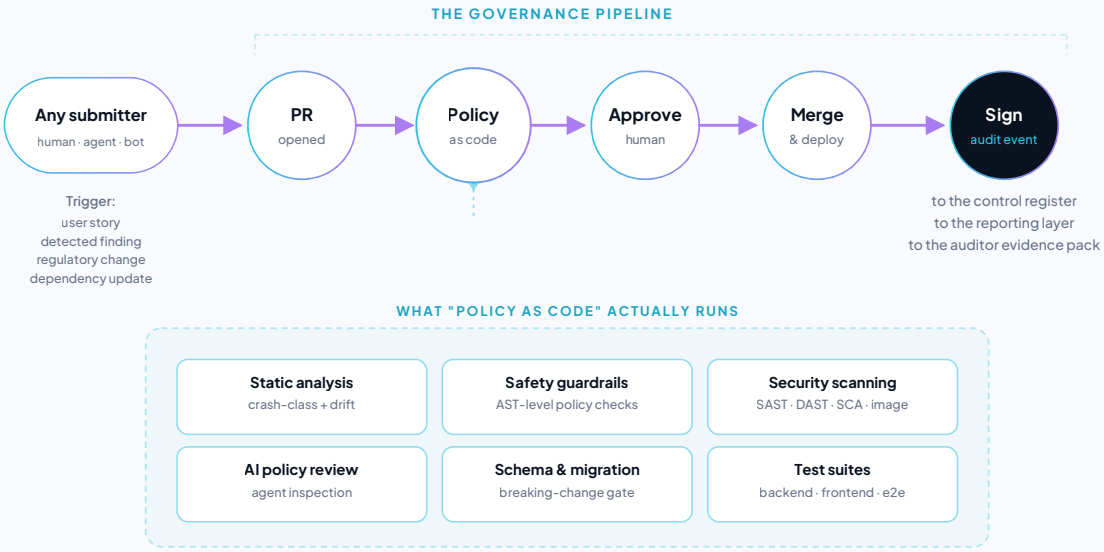
4

Governance at the pipeline, not the submitter

A PR from a human engineer, an AI agent, a contractor, or a dependency bot runs the same policy inspection and the same approval gate. Governance is enforced on the pipeline, not on the person or system submitting to it. This is what makes it robust to AI-generated code, external contributors, and whatever opens a PR five years from now.

What it looks like in practice

Governance as a pipeline runs a small number of stages, under human approval where approval is required, emitting signed evidence at each gate.



Five stages, one signed event trail, submitter-agnostic. Greenfield features, remediation work, regulatory updates, dependency bumps. Every change runs the same gates. ***The Policy stage is a cluster****: six categories of automated check run in parallel before a human sees the pull request at all. The reporting layer, auditor evidence pack, and board view all draw from the event stream on the right.*

THIS IS NOT THEORY

We built it. *ICRG* is the implementation of the pipeline on this page, shipped with an AI-SDLC agent layer underneath.

We wrote the paper because we built the thing. Not the other way around.
xiaotimelabs.ai/icrg.html

Objections we hear, and what we say back

> ***Governance slows development.***
>
> Governance-as-gate slows development. GRCDevSecOps runs at pipeline speed because it is the pipeline. The slowdown everyone has experienced is the slowdown of a weekly-cadence compliance program chasing a daily-cadence engineering program. Same-cadence governance does not slow the cadence.

> ***Compliance is reporting, not engineering.***
>
> That is the attestation-era view. Regulators are increasingly asking for operational evidence, not annual attestation. NIST AI RMF names continuous monitoring as a first-class function. The EU AI Act requires lifecycle risk management. ISO/IEC 42001 calls out ongoing measurement. The reporting-only model is already losing ground in the frameworks themselves.

> ***This sounds like shift-left compliance.***
>
> Shift-left is a direction. GRCDevSecOps is a category. Shift-left says "move compliance earlier." GRCDevSecOps says what that looks like architecturally. Policy-as-code at the PR. Evidence as a byproduct. Remediation through the same pipeline. Submitter-agnostic gates. Those are the nouns that make the verb work.

> ***My auditor still wants a SOC 2 letter.***
>
> Your auditor will still get one. The question is where the evidence in that letter comes from. A platform that produces continuous evidence can render a SOC 2 projection over its control register with less work than a platform that collects SOC 2 evidence quarterly. The letter is the same. The substrate that produces it is different.

> ***We are not ready to enforce policy at the PR.***
>
> Fair. The rollout does not require starting there. Start with a unified control register. Connect the cheapest telemetry. Produce continuous evidence. Then move policy into the pipeline once the evidence base is trusted. The order matters. Policy enforcement without evidence produces compliance theater.

The category bet

In five years, GRC tooling will not look like the current generation of reporting-layer platforms. It will look more like what CI and CD look like for engineering teams today. Pipelines. Policy as code. Event streams. Signed artifacts. Continuous measurement. Remediation loops.

The buyers are already changing. SOC 2 is a table-stakes letter. The conversation in the CISO office is moving to continuous evidence, AI governance obligations, operational risk quantification, and whether the next enterprise security review will take two weeks or two months. The buyers asking those questions are not going to be served by a faster version of quarterly polling.

The organizations that win this shift will be the ones that treat governance as a pipeline problem, not a reporting problem. Some of them will build it. Most will buy it. Either way, the shape is the same.

We think the category is **GRCDevSecOps**. We think it is inevitable. We think the teams that move first will have a meaningful advantage on the first audit that asks "show me the controls operating," and on every enterprise deal whose security review is a closed loop instead of a document exchange.

What to do with this

Read this paper as an argument, not a sales pitch. If it holds up against your environment, you have a hypothesis worth testing. Three practical moves:

1. Audit your current GRC stack against the four principles. For each principle, mark whether you are currently doing it, planning to do it, or not doing it. The gaps are the roadmap.

2. Pick the cheapest telemetry source you are not yet projecting into a register. Version control events. CI pipeline results. Dependency scanning. Identity telemetry. Whichever is closest. Connect it. Project it. See what falls out. Most organizations find three or four controls they thought needed a separate program are already observable.

3. Run a pilot PR gate. One policy. One check. One pull request. Watch how the team reacts. If the pushback is "this slows us down," the check was badly chosen. If the pushback is "why were we doing this in a spreadsheet before," the category is working.

Why we wrote this

We built **ICRG**. Integrated Cyber Risk Governance, with an AI-SDLC pipeline underneath. It is the implementation of the principles in this paper. A unified control register across every framework that applies. Continuous evidence projected from live telemetry, contextualized with FAIR risk quantification and organization-specific tolerance thresholds. Controls tracked by maturity trajectory on the CMMI 0–5 scale, not just pass or fail. Policy-aware gates that run on every pull request regardless of who or what opened it. A signed audit trail that updates the register on every change. Reporting surfaces for boards, auditors, and customer security reviews, all reading from the same event stream.

The build-from-story flow runs end to end. Detect-to-ship remediation shares the same rails by design, with parts of the detection layer live today and parts in active development. This paper exists because we ran into the walls it describes and decided to build through them. We wrote the paper because we built the thing. Not the other way around.

See the product: xiaotimelabs.ai/icrg.html

About Xiaotime Labs

Xiaotime Labs works with mid-market and enterprise teams building AI systems that need governance infrastructure to scale with their product rather than against it. Our focus is organizations whose AI deployments are real enough to be subject to scrutiny and early enough that the right architecture can still be chosen.

Built by **Craig George** and **Stephan Hundley**. Two former CISOs who saw the gap, and how it could be filled. They built ICRG because they needed it.

Contact · info@xiaotimelabs.ai **Learn more** · xiaotimelabs.ai/icrg.html

A technical companion covering architecture and implementation detail is available to prospective customers under NDA.

One more thing

If this paper reads like an argument that should be obvious in two years and contentious today, that is the intent. Category papers only earn their title in retrospect. The teams that move on the argument now are the ones that get to decide what the category looks like.

We built ICRG because we thought that way. You should evaluate whether you want to, too.