

ENGINEERING AN AUTONOMOUS SDLC AT SCALE

Prompt → Prod.

From ad-hoc prompting to orchestrated agentic work.

SPEAKER

Andrew Swerdlow

ROLE

Sr. Director · Engineering
Acceleration & Core Platforms
· Roblox

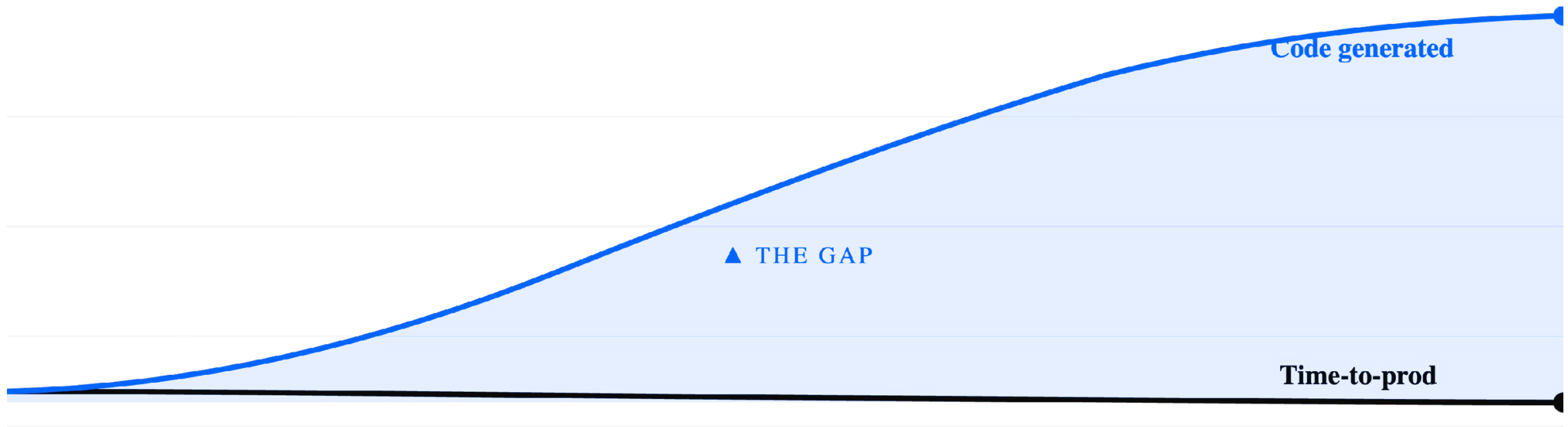
TRACK

AI for Software Engineers

SLOT

Mon · 03:40 PM · Grand
Ballroom

We solved typing. We didn't solve trust.



The bottleneck moved from writing code to reviewing, validating, and trusting it in production

The future isn't better ~~autocomplete.~~

It's Infra systems that can safely **absorb autonomous change** at the scale of a mature codebase.

We need to build Infrastructure for Self-Driving Systems. Everything that follows is about building it.

A reframe. Not an upgrade.

AD HOC

Individual & assistive.

IDE completions. Prompt files. Inconsistent across repos.
Productivity gain is local.



ORCHESTRATED

Multi-agent & governed.

Agentic PRs. Continuous migrations. Workflows committed to
Git. Productivity gain is structural.

Different cognitive posture. Different infrastructure. Different metrics.

Speed and autonomy without safety just creates **debt faster.**

-
- 01 Context & Ownership

 - 02 Reliability & Security

 - 03 Accountability when something breaks at 3am

We need systems that limit the blast radius of autonomy gone wrong

WHAT YOU'LL WALK OUT WITH

Three takeaways

TAKEAWAY · 01

Alignment & Guardrails

Encoding expert judgment as operational systems.

TAKEAWAY · 02

Security & Access

Architecting an agentic SDLC that securely runs on its own.

TAKEAWAY · 03

Measuring What Matters

Productivity metrics for an Agentic native organization.

TAKEAWAY · 01

Alignment & Guardrails

Encoding expert judgment as operational systems.

Mosts company's first reaction to autonomous agents is rejection.

How do we get to "Yes", without compromising prod?

Security said **slow down.**

We had to rethink how to introduce less-controlled agentic frontier harnesses and models.

CONCERN · 01

Prompt-injection → source exfiltration

A poisoned dependency or malicious doc could turn an agent into a data-leak channel.

CONCERN · 02

Inherited user permissions, including prod access

Agents acting as developers would inherit everything developers can do — by default.

CONCERN · 03

The rogue-agent scenario

One compromised context could mean real data deletion risk against production systems.

THE TENSION

Many issues to consider, but the industry was moving fast

We did not want our engineers to fall behind.

Not features. **Agent safe Infra.**

GUARDRAIL · 01

Sandboxing

Blast-radius containment for every agent run. Local + cloud.

GUARDRAIL · 02

Policy gateways

MCP gateway · deep packet inspection · JIT Vault · Least privileged access. No long-lived prod secrets.

GUARDRAIL · 03

Hardening

Agent identities separate from user identities. Always auditable, by design.

Designed to treat agents as untrusted and limit any blast radius

Why autonomous coding can **break** in production.

And what alignment actually means when the model has read the internet — but not your company.

The model
knows the
internet.

It doesn't know your
company.

- Company conventions
- Migration scars
- Security patterns
- SEV lessons
- Infra assumptions
- Ownership graphs

**"Your engineering culture is
not in the training set."**

Not prompting. Not fine-tuning. Operational learning.

WHAT IT IS

Expert judgment captured as scoped, testable exemplars

Config-as-code · versioned · auditable. Each exemplar runs as a focused sub-agent against matching file patterns.

WHY IT MATTERS

The alignment engine — not the model — is the durable asset

The model can change tomorrow. The exemplars stay. Your institutional voice is portable across frontiers.

Codebases are **software** and **memory systems**.

SYMBOLIC

ASTs · dependency graphs · ownership

What calls what. What breaks if I change this. Who's responsible.

Correctness.

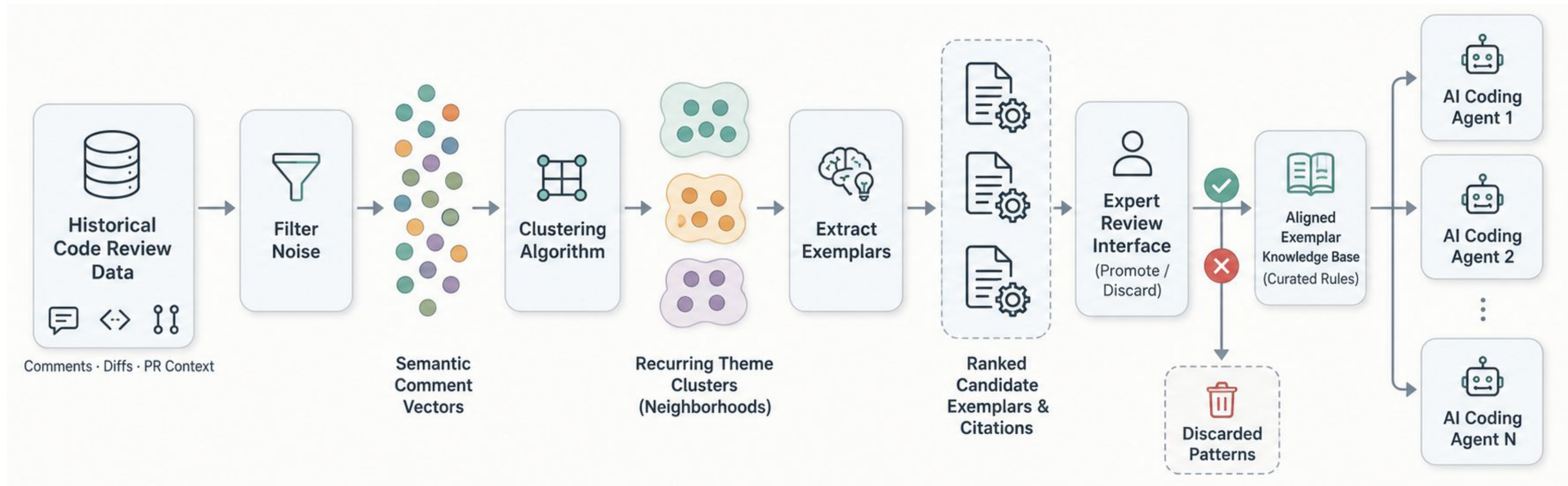
VECTOR

**Embeddings over PRs · reviews ·
exemplars**

What this code **means**. What's similar elsewhere. **Meaning.**

Together — agents reason like a **senior engineer**, not a stranger.

Turning 1.7M review comments into organizational memory.



700K PRs (3 yrs). Deep research over historical reviews → exemplar candidates → expert promote / kill. Expert "no"s become training signal, not tribal knowledge.

A systems based approach to curating deep cultural knowledge.

The alignment engine leverages each exemplar's expert curated scoped guidance and instructions for detecting violations during code review for codebase consistency and quality.

* Exemplar ID

⚠ Changing ID will reset historical metrics (treated as new exemplar)

Category

Performance

Optional - categorize the exemplar for prefixing the review comment on GitHub

* Description & Examples

Blocking inside a high-frequency loop leads to increased latency and thread exhaustion. When a `FetchData` call is made in an async task, warn the author about latency and thread exhaustion. `FetchData` is OK as long as the task has been awaited already. Provide a direct link to async best practices at: [internal_guidance/async](#).

Describe the coding standard this exemplar enforces. Include good and bad code examples where possible. Be as detailed as possible — this prompt is supplied to BuilderAI to detect violations.

File Patterns (one per line)

Glob patterns to control which files this exemplar applies to. Leave empty to apply to all files in the PR. Examples: `**/*.ts`, `src/**/*.py`, `**/*Controller.cs`. Uses [gitignore pattern syntax](#). Use `!` prefix to exclude files (e.g., `!**/*.test.ts`). Exclusion patterns always take precedence over inclusion patterns.

Quick Test Evaluation Suite

Test your exemplar against multiple PRs with expected outcomes to validate precision, recall, and accuracy.

Evaluation Cases + Add Case Run All

Results

Accuracy	Precision	Recall	F1 Score
90%	0%	0%	0%
0 True Pos	0 False Pos	9 True Neg	1 False Neg

Test Case 1 Pass

Pull Request

Expected Outcome

Should find violation Should be clean

Review up to commit (optional)

Test Case 2 Fail

Pull Request

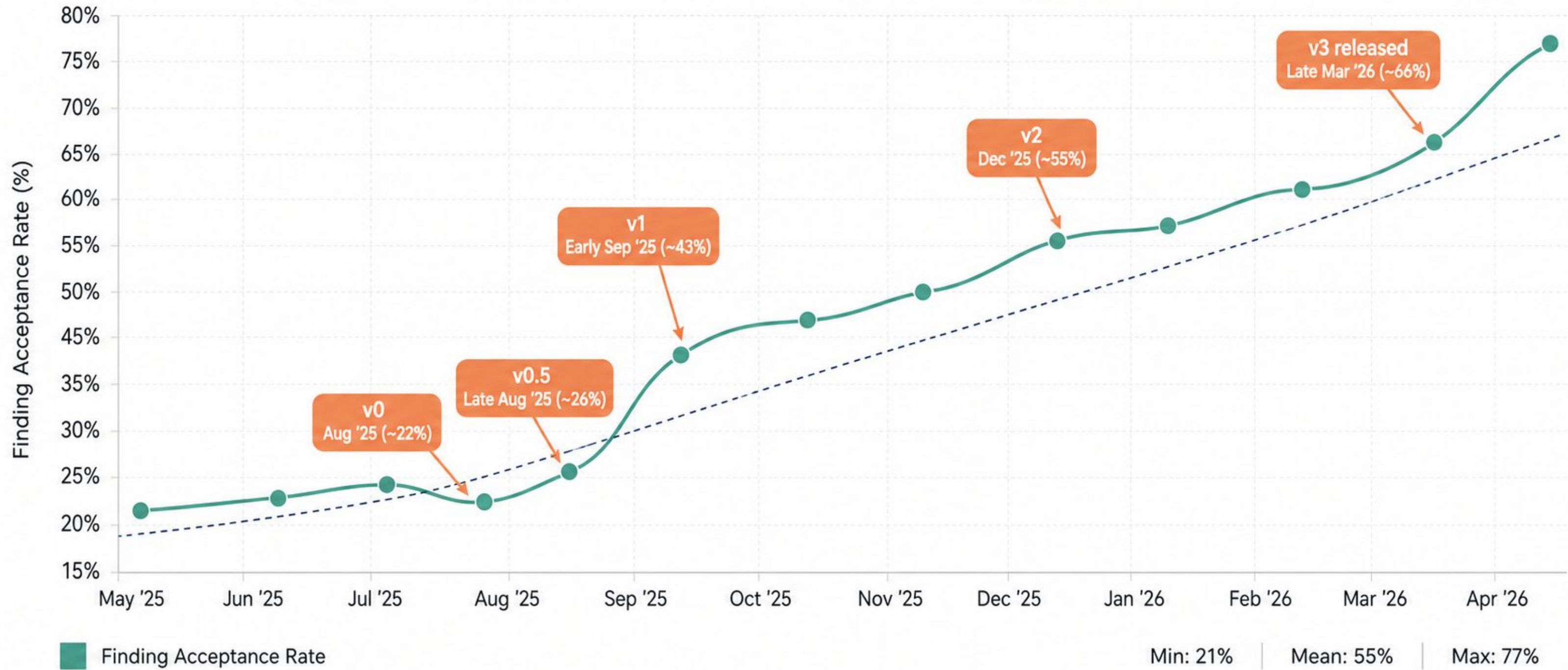
Expected Outcome

Should find violation Should be clean

Review up to commit (optional)

21% → 65%+ accepted findings, weekly.

Finding Acceptance Rate – General Code Review Agent



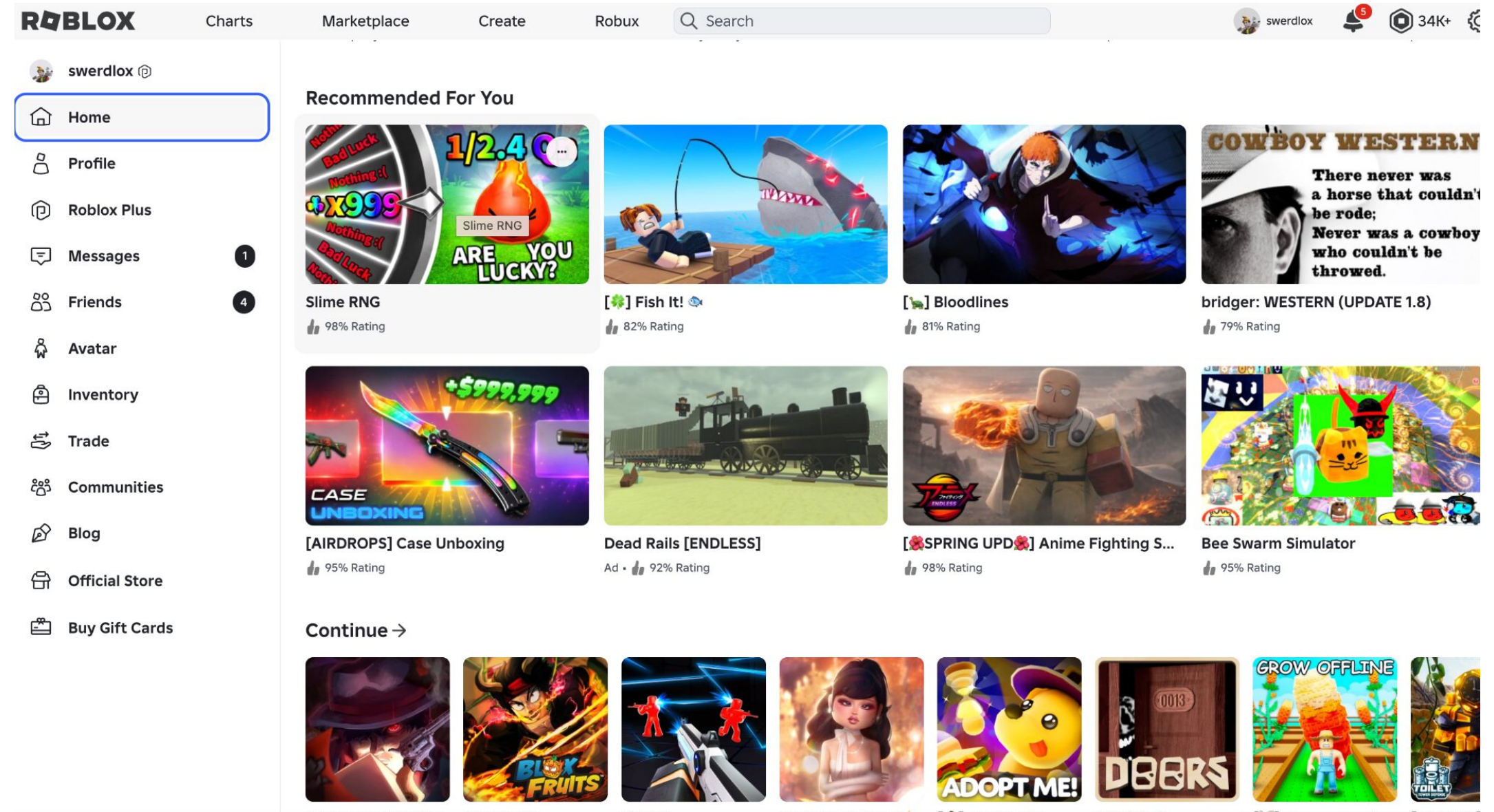
87% LGTM recall . Human review acceptance ≈ 55%... the agent now exceeds it.

TAKEAWAY · 02

Orchestration & Access

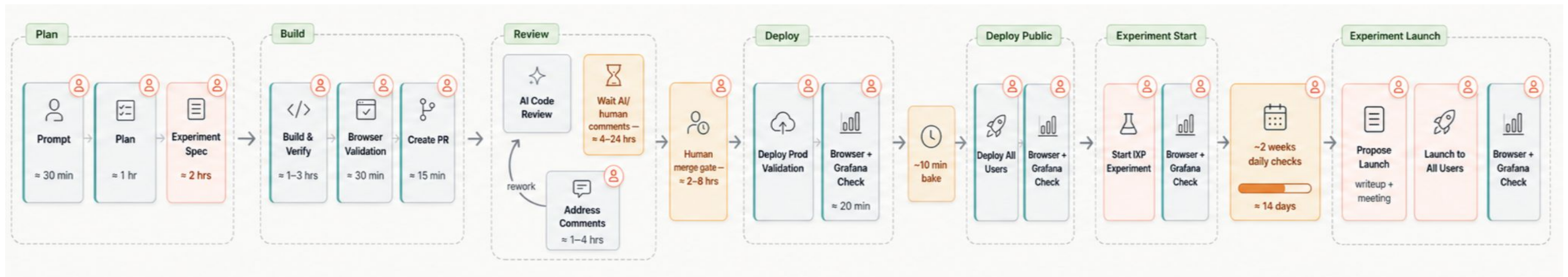
Architecting an agentic SDLC that runs on its own — prompt to production.

**Vision:
Prompt a
change on
the Roblox
home page,
all the way to
production!**



Roblox Homepage 144M Daily Active Users

Old workflow were cumbersome even for humans



TOIL TALLY

 ≈ 18 human touchpoints	 ≈ 3 weeks elapsed	 ≈ 1-2 days active engineer time
--	---	---

Quickly found the gaps

We needed to make sure that Agents had:

Programmatic Access

Many tools were inaccessible to Agents and needs MCPs, CLIs, or APIs.

Safe Usage

We needed safe runtimes and more validation like tests and observability used by the Agents.

Policy Enabled

We had to update policies like Agent access to mutate tools, and change production autonomously.

Programmatic access to every system the workflow touches.

TOOL · 01

Experimentation CLI · start · stop · ramp · fetch results (replaces portal clicks)

TOOL · 02

Web-app deploy CLI · push frontend bundles to staging and production from an agent

TOOL · 03

Browser-automation CLI · headless UI validation inside the agent's verification loop

TOOL · 04

Telemetry CLI · query production metrics during rollout to confirm no regressions

TOOL · 05

Pull-request CLI · open PRs, attach artifacts, iterate on review-agent feedback

Agents:

- Write
- Review
- Fix
- Approve
- Deploy
- Iterate

The screenshot displays a code review interface for BuilderAI. At the top, it indicates 'BuilderAI reviewed last week' with a 'View reviewed changes' link. A comment from BuilderAI (Builder AI) is shown, featuring a 'Tip' section with a lightbulb icon. The tip text reads: 'Use 👍 🙌 emojis to indicate helpfulness - BuilderAI automatically learns from feedback. If a comment is not helpful or inaccurate, please reply to that comment thread with your reasoning to help improve future recommendations. Learn more about BuilderAI Code Review agent [here](#)'. Below the tip, a note states 'BuilderAI can make mistakes, please review the changes thoroughly'. The main part of the review shows a code diff for the file '...oblox.BEDEV2.RuntimeConfiguration/RuntimeConfigurationMonitor/RuntimeConfigurationMonitor.cs'. The diff highlights lines 114 to 117 with a green background, showing the following code changes: line 114: '.UpdateCachedProjectGroupConfigurationAsync(args.ConfigBuilderMetadata, sh', line 115: '.ConfigureAwait(false)', line 116: '.GetAwaiter()', and line 117: '.GetResult();'. At the bottom, another comment from BuilderAI (Builder AI) is partially visible, dated 'last week', with a 'Performance: Blocking on an async call via GetAwaiter().GetResult() when updating cached project group'.

TAKEAWAY · 03

Measuring What Matters

Productivity metrics for an Agentic native organization.

Most teams measure the **wrong** thing.

VANITY · OUTPUT

~~Tokens · completions · suggestions~~

~~Lines of code generated~~

~~PRs opened~~

~~Engineers "using AI"~~

SIGNAL · OUTCOME

Agent quality

Are the agents producing high quality results

Rollback & incident rate

the cost of being wrong

Feature velocity

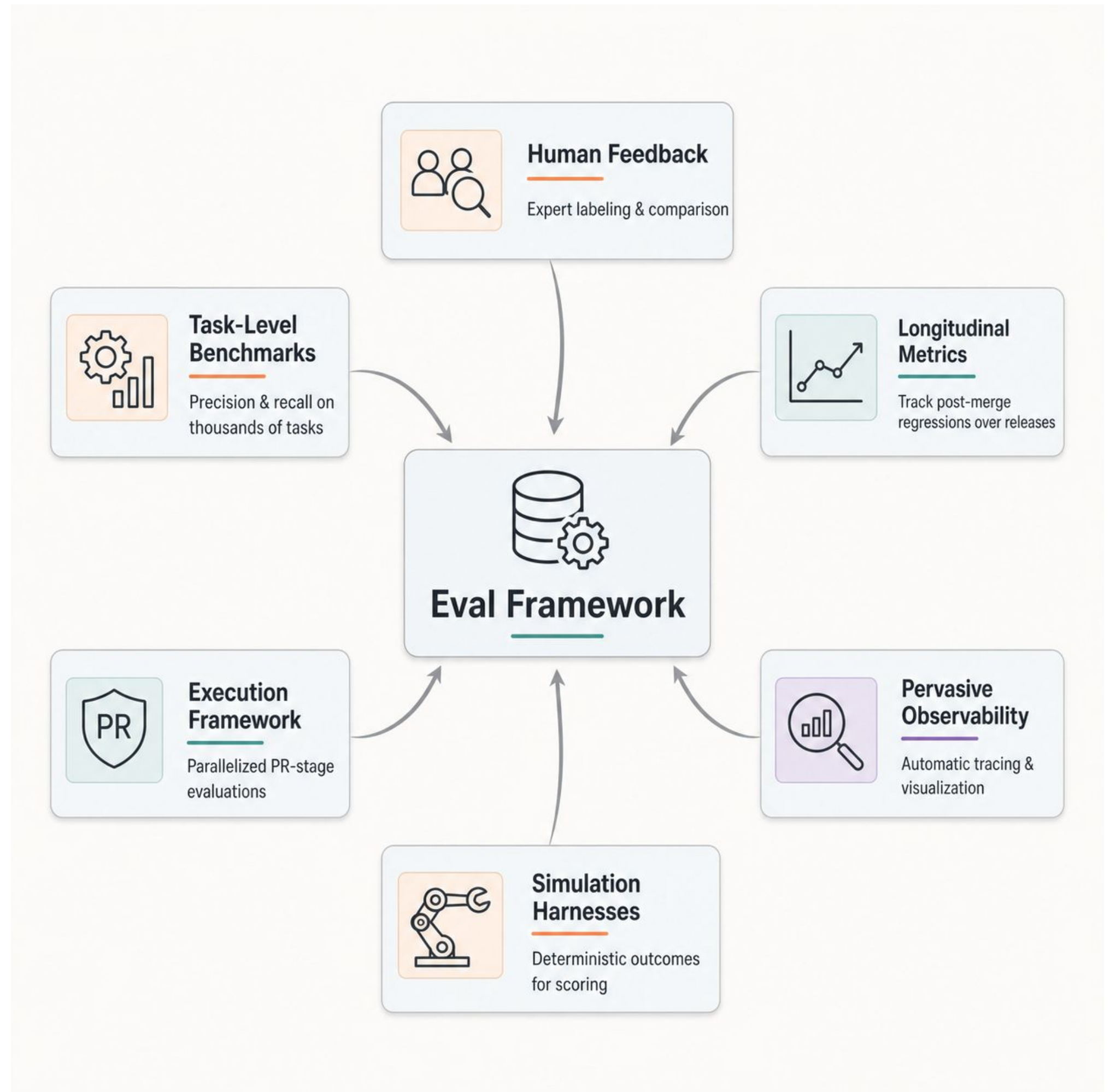
delivered behavior, not authored text

Autonomous-completion %

work done without a human in the diff

Measure what **ships** — not what is *generated*.

Agent quality matters.



Feature Velocity

Feature Velocity

Breakdown and trends of merged PRs by category over the last 6 months.

Last 6 Months ▾

Aggregation: Auto ▾



PRS ANALYZED

174.6k

across the last 6 months



MEDIAN PRS / ENG

27

monthly median per engineer

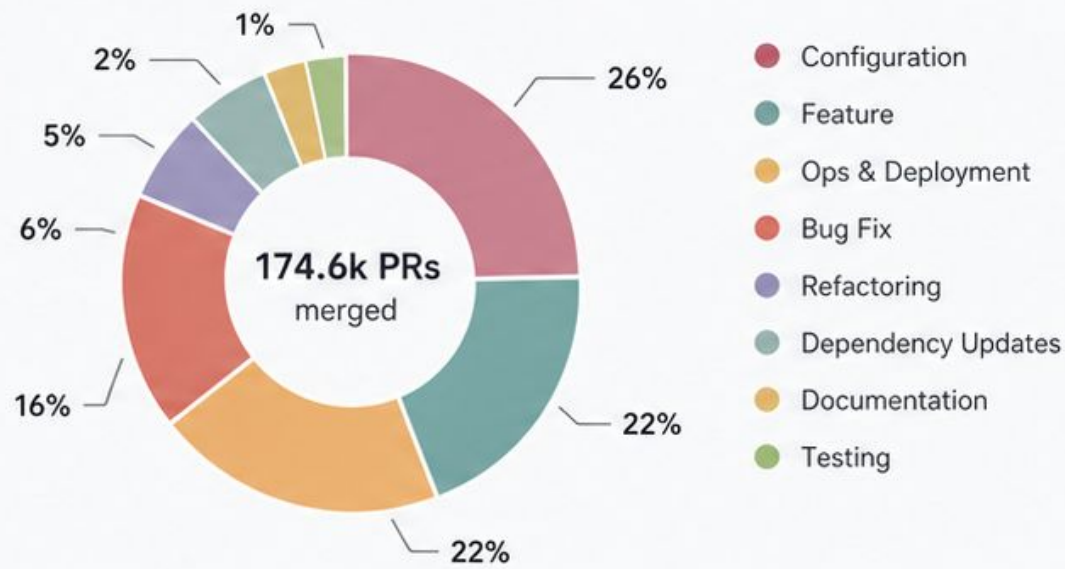


MAR PEAK

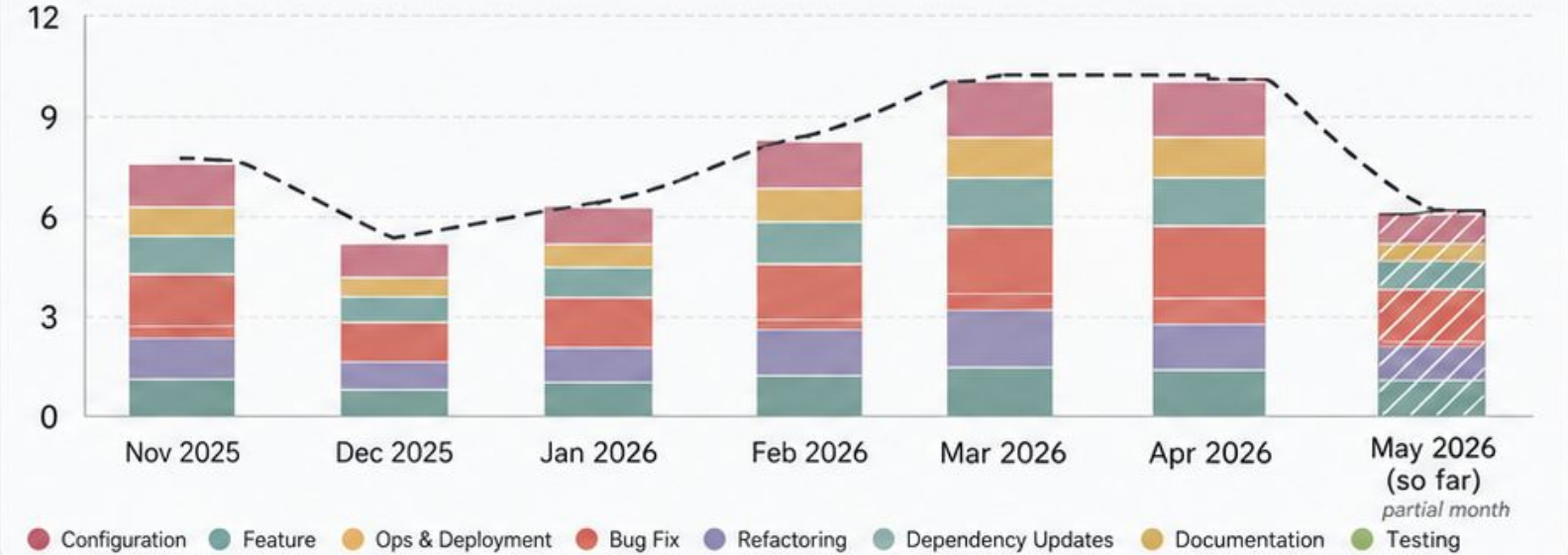
10

median PRs/eng, **+43%** vs Nov

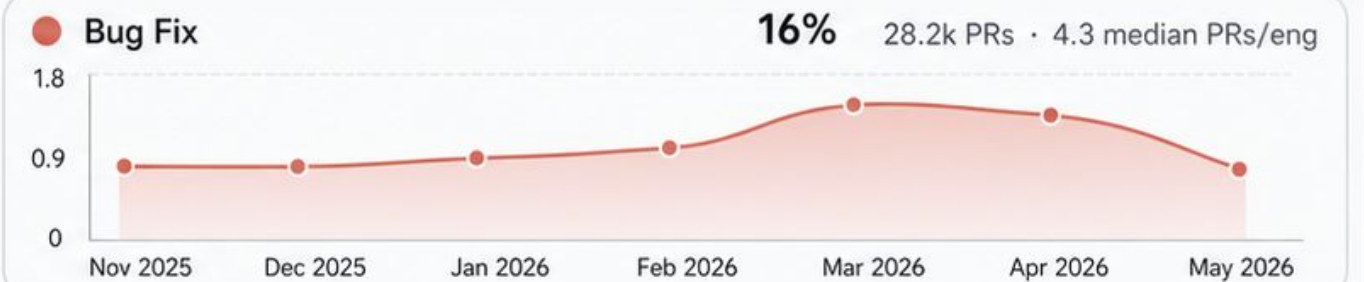
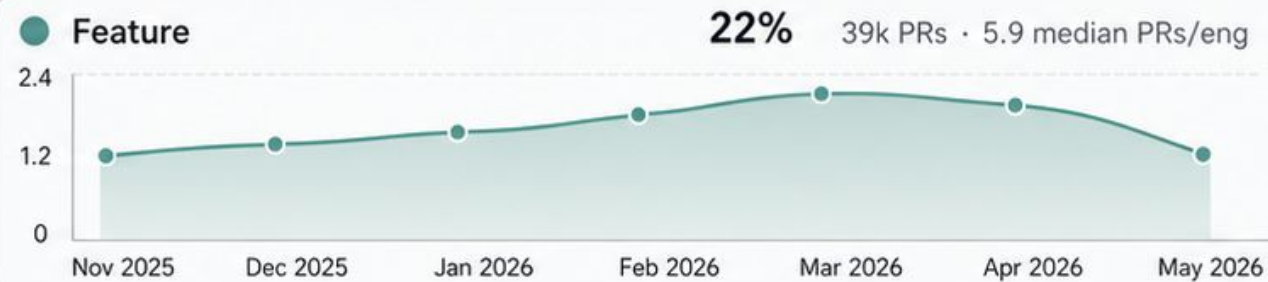
Work Distribution of Merged PRs



Trends Over Time



Per-category breakdown



Agent Autonomy

Are we getting more autonomous?

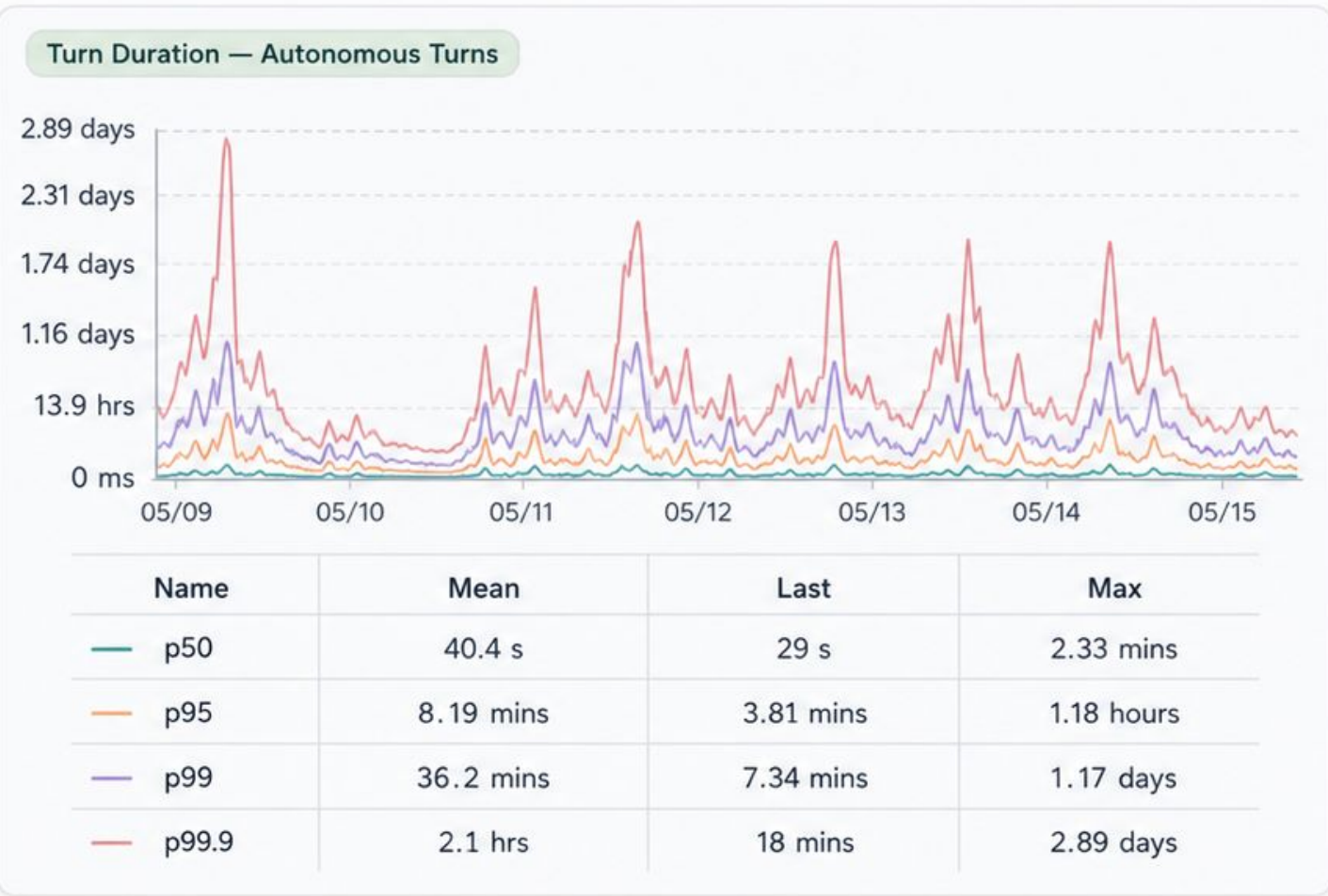
Longer agent turns = less human intervention. Error contributors are what break the chain.

P50 TURN
40.4 s
median run between human touch

P95 TURN
8.19 min
95% of turns finish within

P99 TURN
36.2 min
tail of long autonomous runs

P99.9 MAX
2.89 days
longest unattended run



limits →

Top Error Contributors
What's interrupting autonomous runs

Error	Count
Request was aborted.	1,496
Ripgrep search timed out after 20s.	1,027
Streamable HTTP error: invalid...	676
Tool response is too large to return.	536
401 invalid token: signature expired	458
The operation timed out.	375

Top Runtime Versions by Error Rate
Where errors concentrate

Version	Errors	Error Rate
2.1.126	5,083	13.7%
2.1.112	4,278	15.7%
2.1.132	1,566	8.14%
2.1.123	1,127	17.7%
2.1.128	1,094	37.5%
2.1.119	645	9.15%

Goal: extend turn duration by eliminating the top error contributors

WHAT I HOPE YOU'LL WALK AWAY WITH

Three takeaways on the way to 24x7 AI

TAKEAWAY · 01

Alignment & Guardrails

Encoding expert judgment as operational systems.

TAKEAWAY · 02

Security & Access

Architecting an agentic SDLC that securely runs on its own

TAKEAWAY · 03

Measuring What Matters

Productivity metrics for an Agentic native organization.

INSTRUMENT · CURATE · GOVERN · TRUST

Thank you.

SPEAKER

Andrew Swerdlow
Sr. Director · Roblox

FIND ME

[linkedin.com/in/andrewswerdlow](https://www.linkedin.com/in/andrewswerdlow)
andrewswerdlow.net

BOOK

[Tech Leadership:
The Blueprint for Evolving from Individual Contributor to Tech Leader](#)