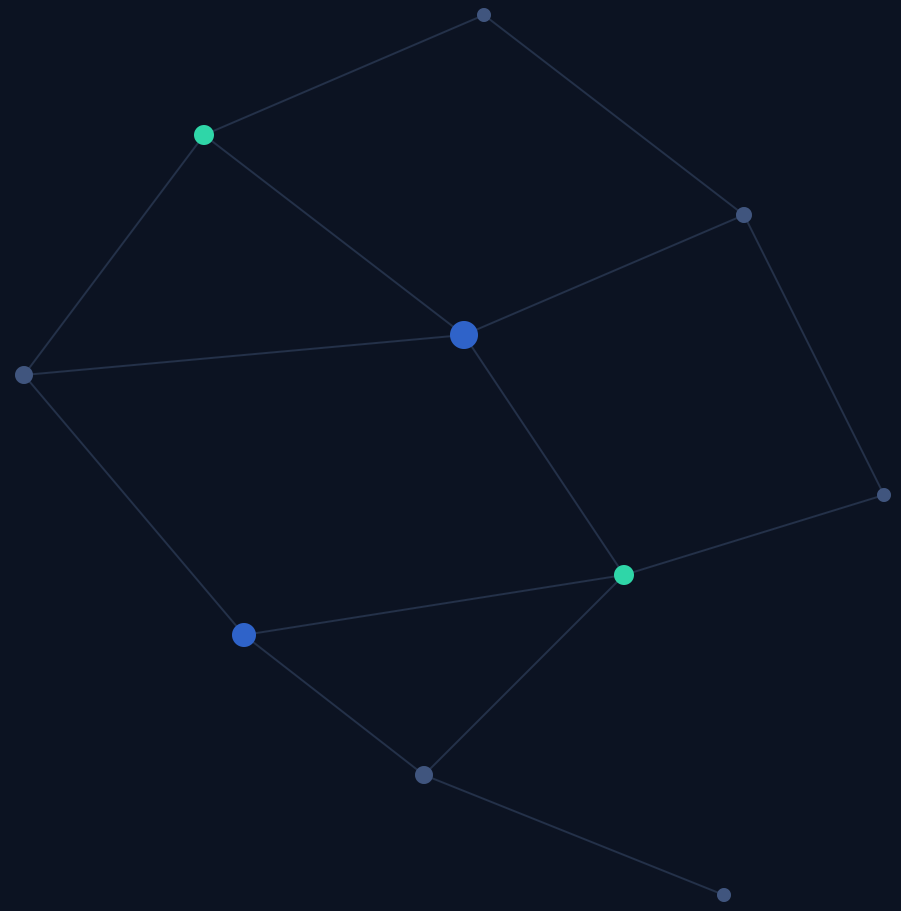
PlayerZero

CAPABILITIES OVERVIEW · FOR ENGINEERING LEADERSHIP

The AI production engineer for complex software systems.

PlayerZero builds a living model of how your software actually works — then puts agents to work on triage, debugging, code review, and QA, with your engineers in control.



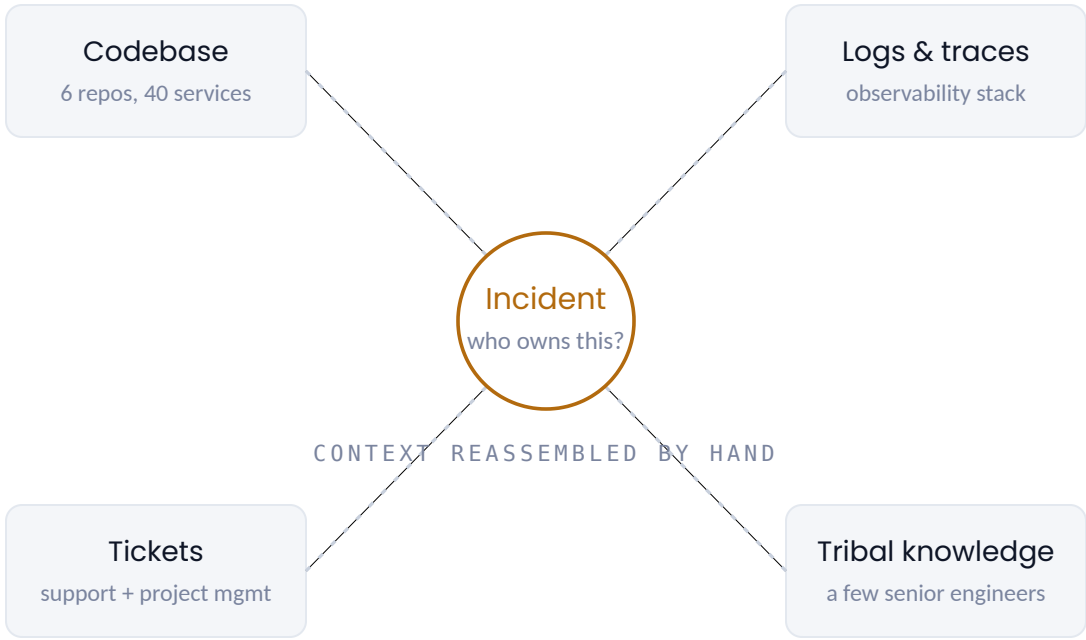
SRE · SUPPORT · QA — ON AUTOPILOT, WITH APPROVAL GATES

Production knowledge is scattered. Resolution pays the price.

The truth about how a system behaves lives in five places at once: the code, the logs, the tickets, the dashboards — and the heads of a few senior engineers. When something breaks, resolving it means reassembling that context by hand, across teams, every single time.

Every handoff — support to engineering, engineering to QA — drops context and restarts the investigation. The result: escalations queue up against your best people, and customers wait.

- Handoffs restart investigations
- Senior engineers stuck on triage
- Same incidents, rediscovered



Four taxes every engineering org pays for scattered context

TIME

Triage is measured in days

Reproducing an issue, locating the owning code, and gathering evidence happens serially, across tools and teams — before anyone starts fixing anything.

FOCUS

Escalations land on your best people

The engineers who can resolve anything become the bottleneck for everything — pulled off roadmap work to answer "what changed?" one more time.

QUALITY

Defects escape to customers

Reviews and tests check what the team thought to check. Impact that spans services, contributors, or releases is exactly what manual review misses.

MEMORY

Institutional knowledge walks out the door

Every resolved incident teaches the org something — and most orgs have nowhere durable to put it. The next on-call rediscovers it from scratch.

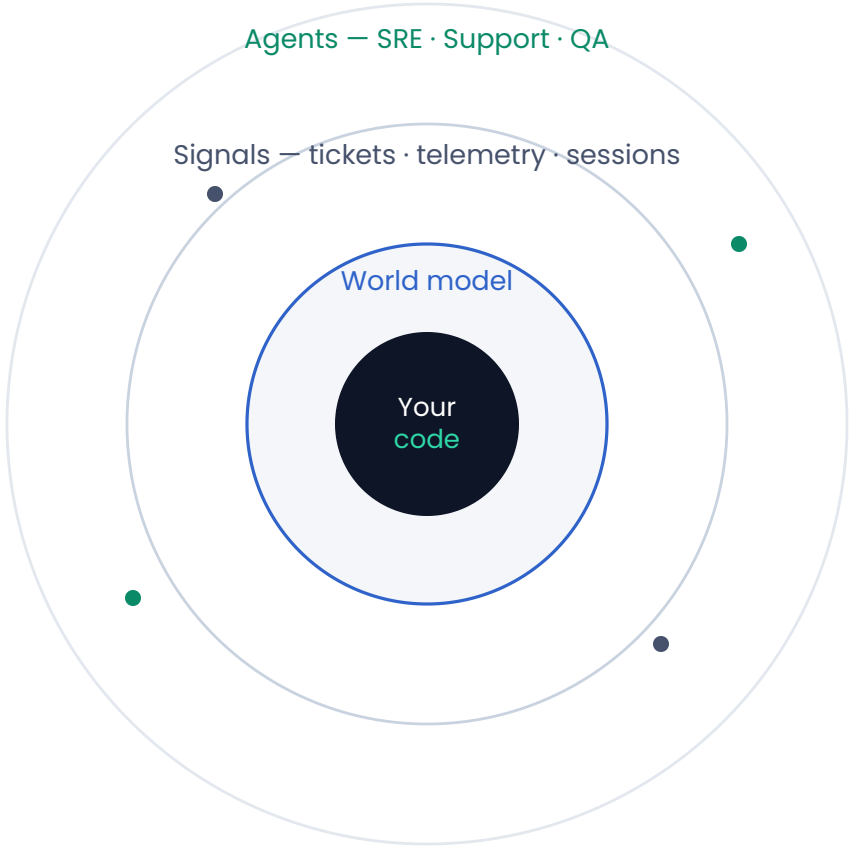
These compound: slower triage means more escalations, which means less roadmap capacity — and none of it scales with headcount.

An AI production engineer, built on a living model of your software

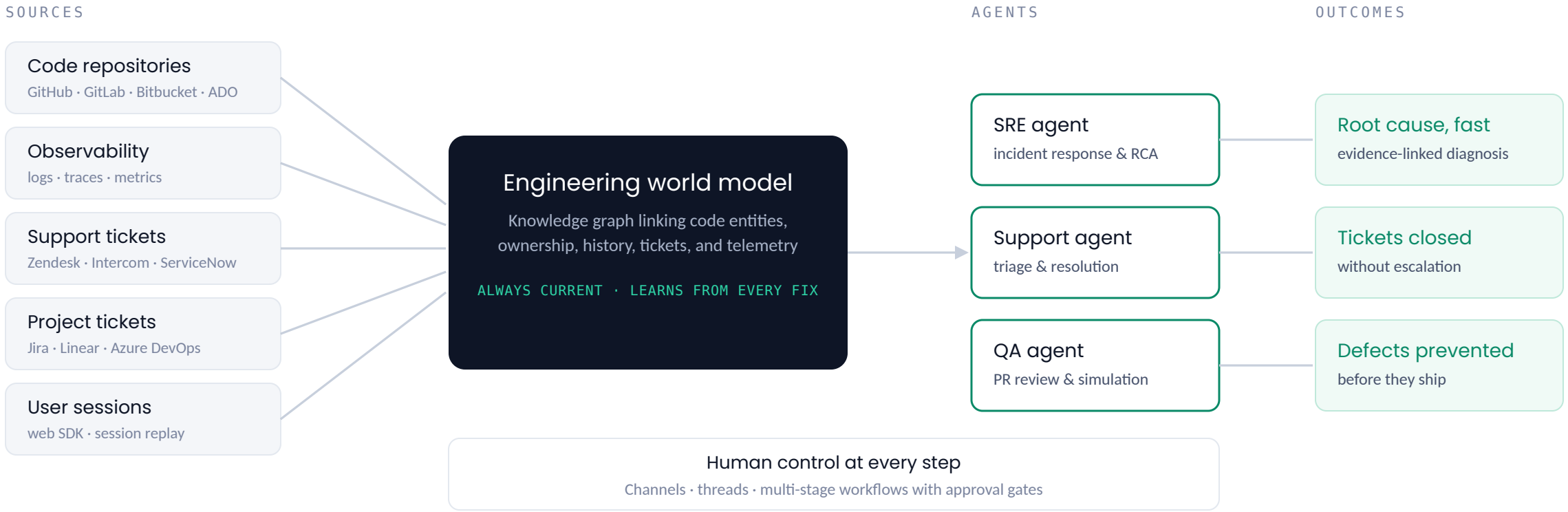
Grounded in code. Every defect, bottleneck, and user issue ties back to the codebase — so PlayerZero anchors everything there. Tickets, telemetry, and sessions enrich the code model instead of living beside it.

Continuously current. Every commit, ticket, and telemetry event refreshes the model. It reflects the system you run today, not the architecture diagram from last year.

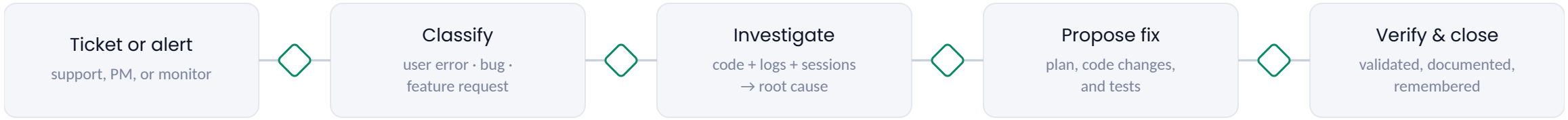
Compounding. Each resolved incident and validated change is captured and reused. The platform gets better at your system the longer it runs — institutional memory that doesn't leave when people do.



One model of the system. Every team works from it.



From "a customer hit something" to root cause — without the war room



◇ = APPROVAL GATE — AUTO OR HUMAN-REVIEWED, YOUR CHOICE PER STAGE

Knows when to ask

Agents are configured to raise a hand instead of guessing — on security, billing, data-loss, or ambiguous-routing cases, a human gets the question first.

Evidence, not vibes

Every conclusion links to the commits, log lines, and sessions behind it. Reviewers audit the trail, not the claim.

Parallel investigations

Hive mode runs specialist agents concurrently on complex incidents — multiple angles explored at once, one coordinated answer.

The same world model, applied before the bug exists

PLAN

Scoping from a ticket

Turns a feature request into an implementation plan: where it should live, what existing code to reuse, ordered steps with effort, risk, edge cases, and tests.

BUILD

Change analysis

Analyzes code and changes in the context of the whole system — dependencies, ownership, and the history of what broke before.

REVIEW

PR vs. acceptance criteria

Maps every acceptance criterion to the code that satisfies it. Flags out-of-scope changes, missing tests, and unimplemented criteria — with cited evidence.

SHIP

Regression detection

Watches releases against the model of what the system did before — catching behavioral drift that unit tests were never written to see.

WHY IT'S DIFFERENT

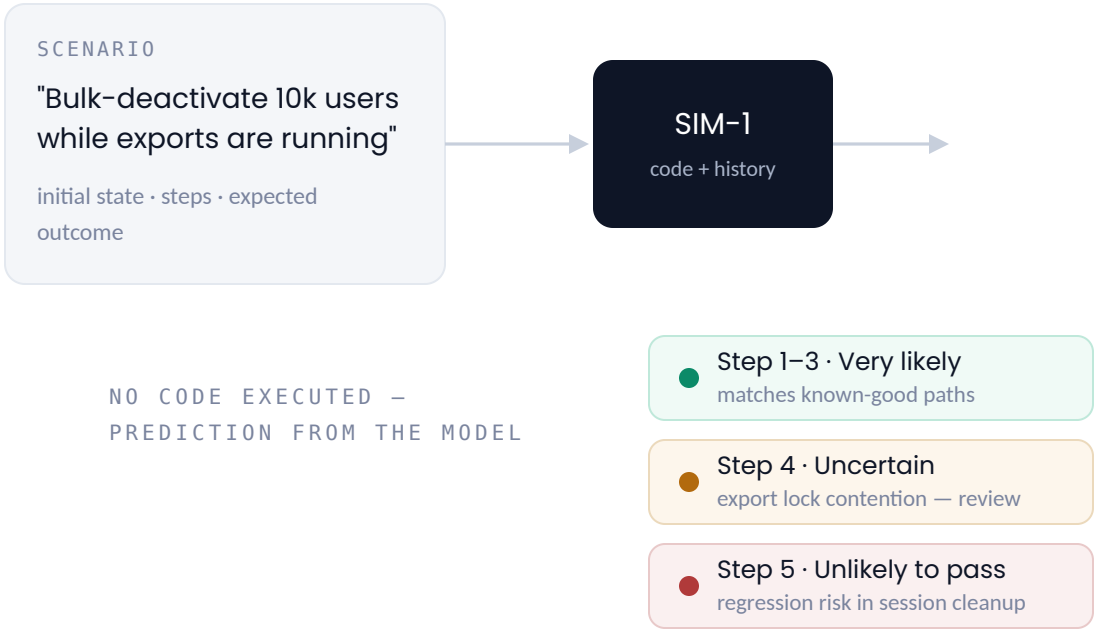
Generic code assistants see the diff. PlayerZero sees the diff *plus* the tickets, the telemetry, and every past incident touching that code — so review verifies intent, not just syntax.

Testing before testing: predict the blast radius of a change

Describe a scenario in plain language — setup, steps, expected outcome. PlayerZero's simulation models (SIM-1) evaluate it against your codebase, dependency graph, and historical failure patterns — **without executing anything**.

Each step gets a risk-rated prediction. Teams find edge cases and gaps before committing to test automation or release — and focus real testing where risk actually concentrates.

- UI flows — validate interactions pre-build
- Backend — model API calls & error handling
- Data — predict persistence & transactions



Living documentation — and a learning loop that compounds

Architecture answers on demand

"How does auth flow through these services?" answered from the current code — with diagrams generated on the spot, not last year's wiki page.

Release notes, generated

Executive summaries, internal functional reports, and customer-facing notes produced from the tickets and code that actually shipped.

Onboarding, accelerated

New engineers ask the system instead of interrupting the team — and get answers grounded in the codebase, with citations.

Skills: expertise, packaged

Your best engineers' playbooks become reusable skills any agent — and any teammate — can invoke.

Agents with guardrails: you define the pipeline, you own the gates

Every agent runs inside a workflow you design — stages, permitted transitions, and an approval policy per step. Autonomy is a dial, not a switch.

Auto- or human-approval, per stage

Let routine classification advance automatically; require sign-off before any fix ships. Change the policy anytime.

Model tiers per stage

Economy models for routine steps, premium reasoning where it matters — or auto-optimize. Cost control is built into the pipeline, not bolted on.

Workflow inbox

One queue of everything awaiting review, grouped by channel. Approve, give feedback, or redirect without switching tools.

Versioned & auditable

Workflows are versioned with rollback. Every thread records what the agent did, saw, and concluded — a full audit trail.

TEMPLATES INCLUDED L1 Support · Engineering Scoping · SRE Alert Response · QA Validation — start from proven pipelines, customize with interactive guidance.

Meets your stack where it is

SOURCE CONTROL

- GitHub
- GitLab
- Bitbucket
- Azure DevOps

PROJECT MANAGEMENT

- Jira
- Linear
- Azure DevOps

CUSTOMER SUPPORT

- Zendesk
- Intercom
- Freshdesk
- ServiceNow

COLLABORATION

- Slack
- Microsoft Teams
- Confluence

CRM & GTM

- Salesforce
- HubSpot

TELEMETRY & SESSIONS

- Observability tools
- Web SDK
- Chrome extension

OPEN BY DESIGN MCP support connects PlayerZero to external AI tools and any internal system — and API triggers let your pipelines invoke PlayerZero workflows programmatically.

What customers report

8×
faster incident resolution

95%
reduction in escalations

84%
reduction in defect escapes

days → min
triage time, before and after

Outcomes reported by PlayerZero customers (playerzero.ai).

TRUSTED BY ENGINEERING TEAMS AT

Verizon | Zuora | Nylas | Georgia Pacific | KeyData | Onboard

WHAT THAT BUYS YOU

Roadmap capacity back — senior engineers stop paying the triage tax

Faster customer answers — support resolves without waiting on engineering

Fewer bad releases — impact caught before customers find it

Built for organizations that can't compromise

SOC 2 Type II

Independently audited controls over security, availability, and confidentiality.

HIPAA

Compliant handling for teams operating in healthcare environments.

ISO 42001

Certified AI management system — governance for the AI itself, not just the data.

Your data, your boundary

Private data storage in your own S3 buckets; `.pzignore` excludes sensitive files from analysis entirely.

Enterprise access control

SSO via Okta and Microsoft Entra (OIDC), org- and project-level membership, PII thresholds and filtering on captured sessions.

PRIVACY PHILOSOPHY Capture what diagnosis needs, exclude what it doesn't — with the controls in your hands, from session-capture thresholds down to individual files.

PlayerZero

NEXT STEP

See it on your codebase.

The fastest way to evaluate PlayerZero is against your own repos and your own tickets. A pilot takes days, not quarters.

STEP 1 — CONNECT

Repos + ticketing, in minutes

Read-only connections to source control and your ticketing system. The world model builds itself.

STEP 2 — PROVE

Run it on real history

Point triage and PR review at your recent tickets and pull requests. Compare its answers to what your team found.

STEP 3 — SCALE

Roll out workflows

Adopt templates for support, SRE, and QA — tightening or loosening approval gates as trust builds.