GitLab

Why coding 10× faster has not made software delivery 10× faster

# The AI Paradox

## From faster coding to governed, agentic delivery



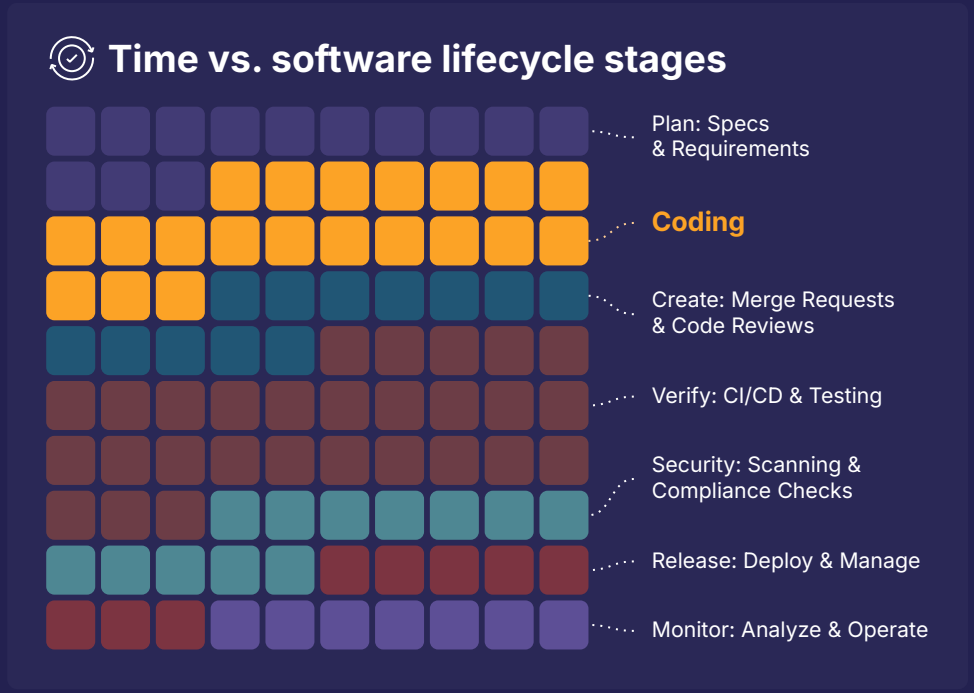
Mobia Connect | GitLab Duo Agent Platform | Vishal Thenge - Solutions Architect

# Coding 10× faster did not make delivery 10× faster

**The bottleneck moved from writing code to coordinating context, quality, risk, and handoffs.**

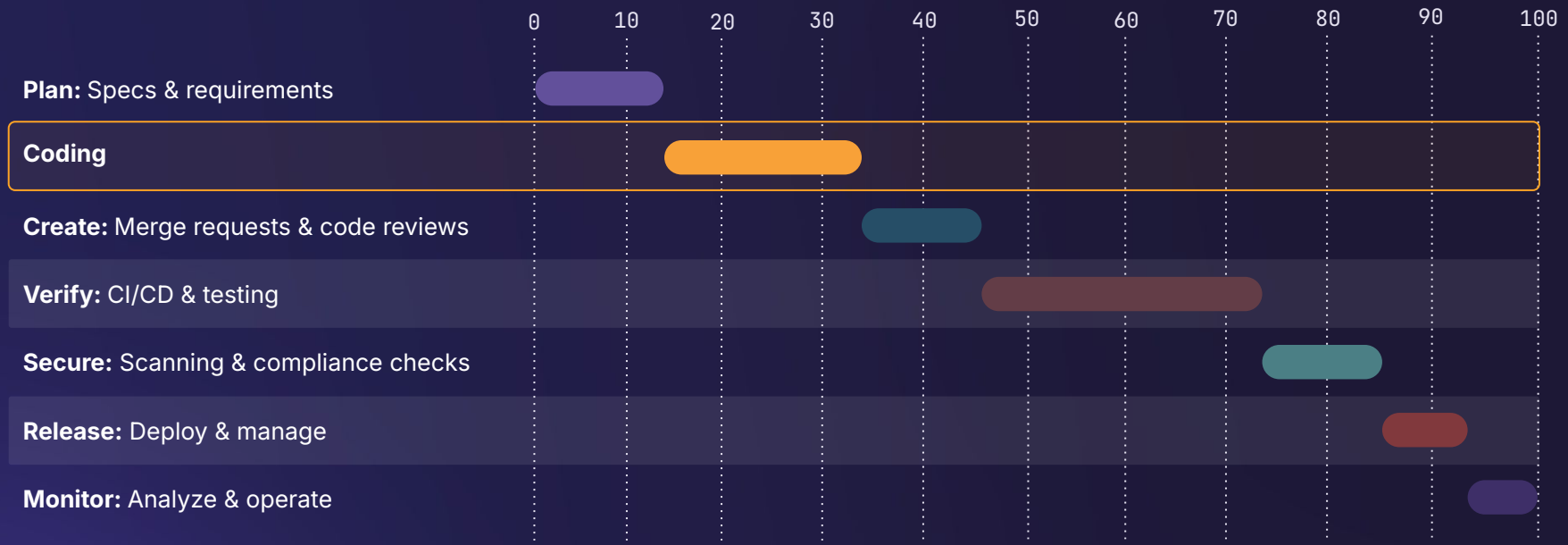
AI accelerated code creation. Delivery still runs through planning, review, testing, security, deployment, and operations.

**The next advantage: organizational delivery velocity.**



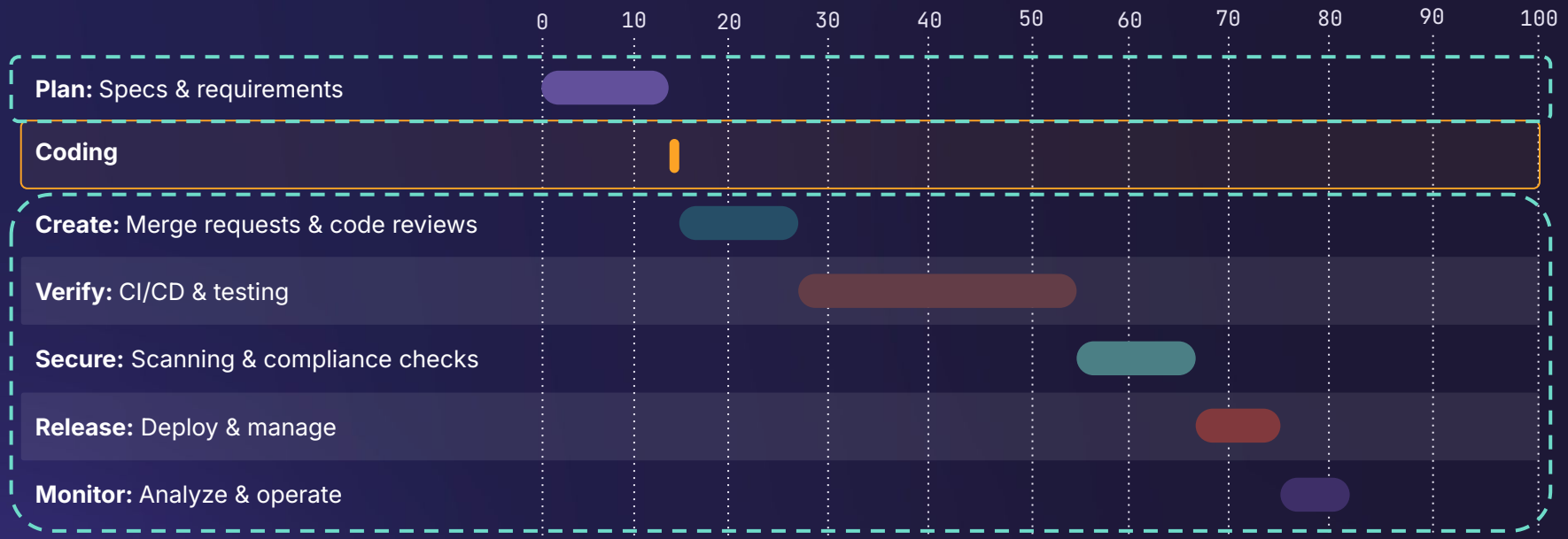

© GitLab Inc.

# The typical software project




© GitLab Inc.

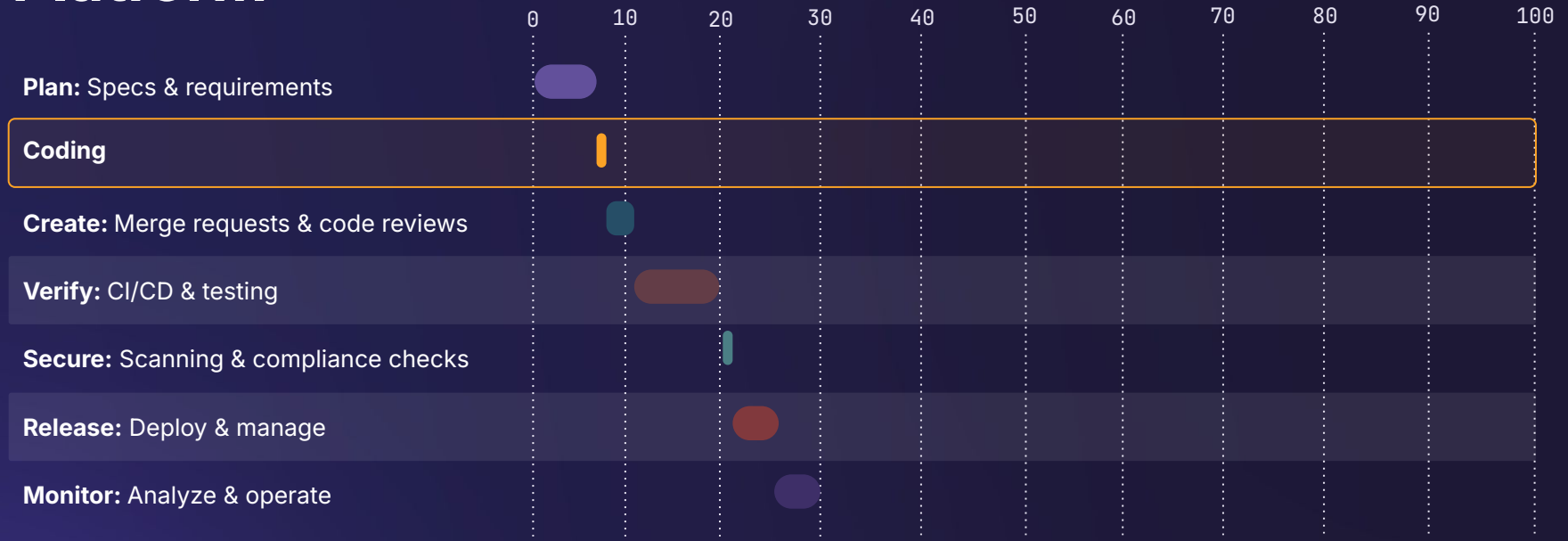
# The bottleneck is the rest of the SDLC




**AI shrank coding. Planning, review, verification, security, release, and operations still set the pace.**

© GitLab Inc.

# Project with AI coding & GitLab Duo Agent Platform



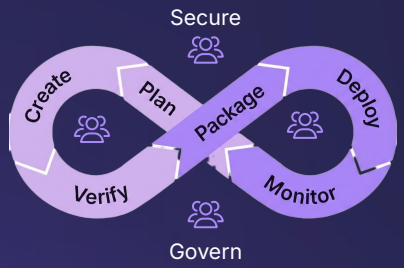

© GitLab Inc.

# Engineering is moving through three modes

Modes 2 and 3 build on the platform foundation customers already have

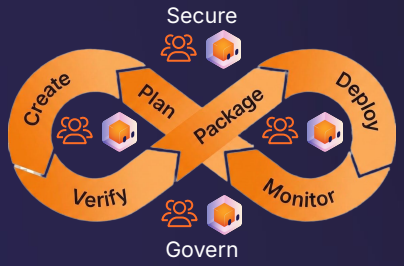
## Mode 1: Manual

People manage every task and handoff.



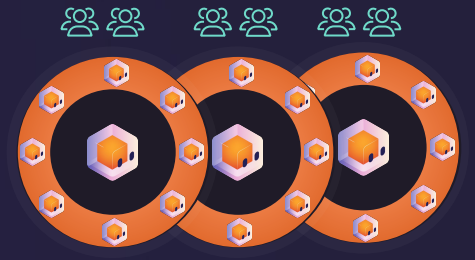

## Mode 2: Agentic Assist

Teams orchestrate agents across the lifecycle.




## Mode 3: Autonomous

Teams define intent; agents plan, execute, verify, and adjust within guardrails.



WEEKS → DAYS → HOURS/MINUTES

© GitLab Inc.

# DAP turns AI assistance into delivery execution

- **Cloud Neutral, Model Neutral**

## How DAP works

- Agents: specialized reasoning and actions
- Flows: multi-step, event-driven workflows
- Unified data model: context across the SDLC
- Guardrails: access, actions, models, environments
- From “help me write code” to “help me complete governed delivery work”




© GitLab Inc.

# The AI Control plane

Four layers that only matter in combination

| | | |
|---|---|---|
| 4 | **Agent & workflow orchestration** | Duo Agent Platform |
| 3 | **Governance & security** | AI Governance, policy, audit |
| 2 | **Lifecycle context graph** | Orbit |
| 1 | **Machine-scale SCM** | Next-generation SCM |

**One platform for every mode**

- **Human-driven**
- **Agentic assist**
- **Autonomous**

**Bottom line:** one control plane for human-driven, agentic-assist, and autonomous engineering.

© GitLab Inc.

# Next-generation SCM for agent-scale work

Git-compatible SCM built for machine-scale engineering, not “replace Git”

PRIVATE BETA / EARLY ACCESS

## Classic Git workflow

Every agent clones the full repository

*full clone, every time*

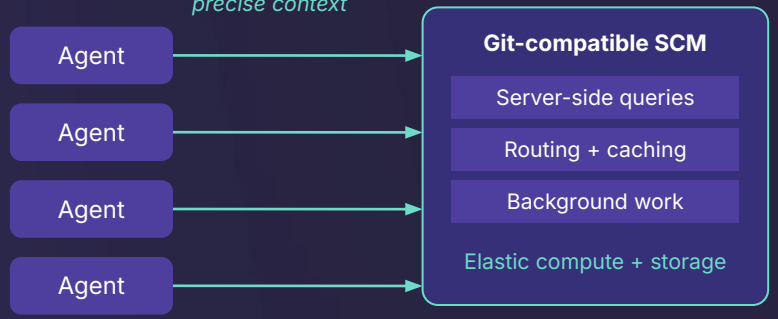



Clone tax | Concurrency pressure | Context overhead

## Agent-optimized access

Agents query the server for precise context

*precise context*




Git protocol unchanged | Precise context | Elastic concurrency

As agent usage scales, the backend, not Git compatibility, is what has to change.

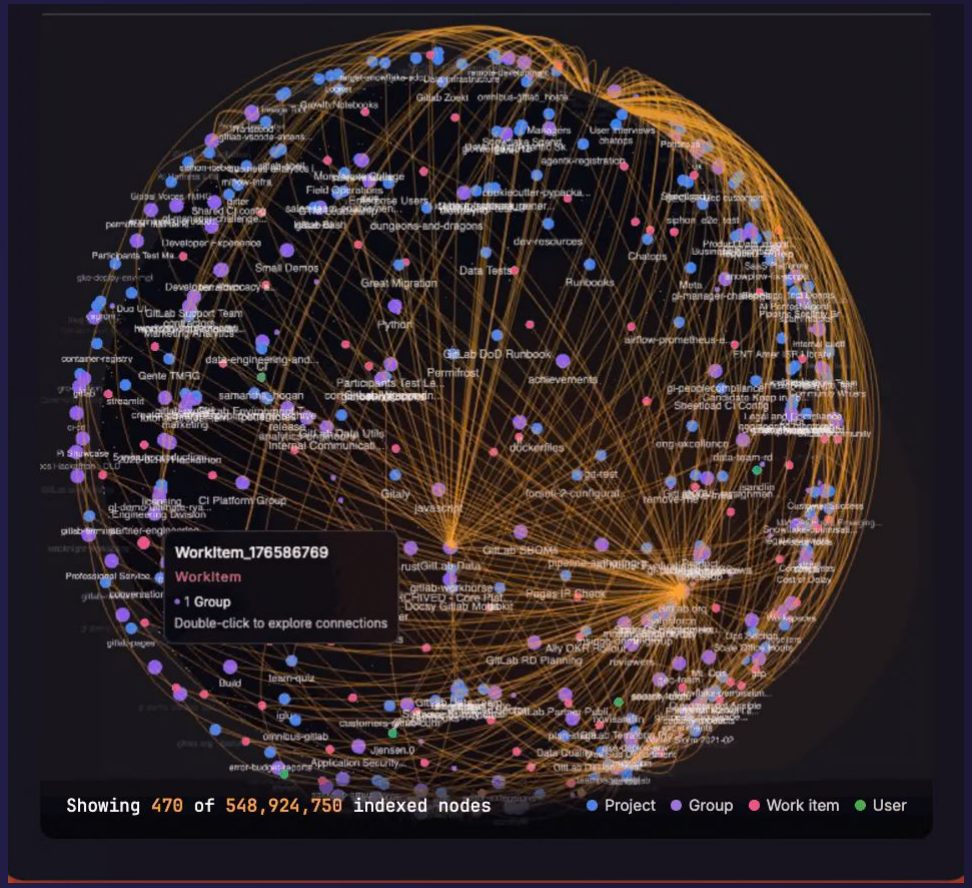
© GitLab Inc.

Public Beta

# GitLab Orbit

## A live map of the software lifecycle

- Continuously updated lifecycle context graph
- Connects code, work items, MRs, pipelines, deployments, vulnerabilities, and ownership
- People and agents query relationships instead of stitching together APIs
- Permissions remain enforced at the graph




© GitLab Inc.

# Orbit changes agents from guessing to reasoning

Without lifecycle context, an agent only sees the file or task in front of it

DEPENDENCIES

**What depends on this change?**

RISK

**What security or deployment risk is connected to it?**

OWNERSHIP

**Who owns the affected system?**




**With Orbit, agents reason across relationships:** less context starvation, fewer hallucinations, better impact analysis, lesser tokens consumed

© GitLab Inc.

# Governance is what makes agentic scale possible

Speed with control: governance is the accelerator, not the blocker

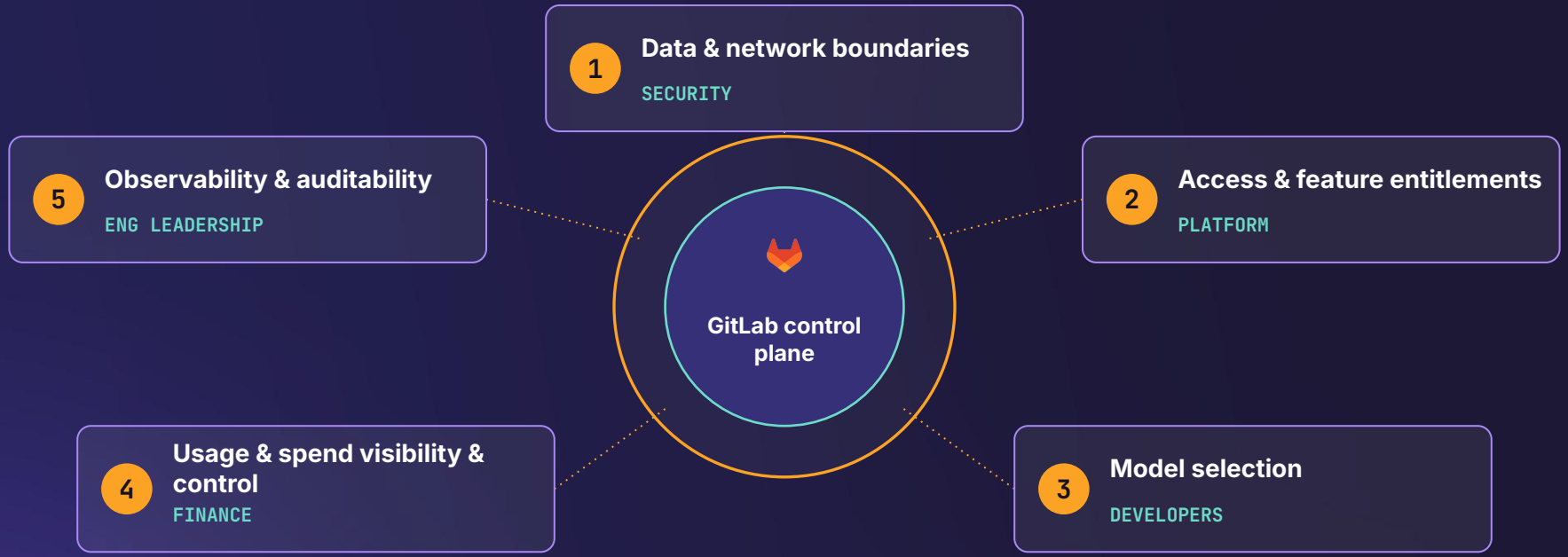



1. **Agents spread faster** than governance does
2. **Security & compliance** need proof of what agents accessed and changed
3. **Platform teams** need central policy, not stitched point controls
4. **No control plane,** no confident path from pilot to scale

© GitLab Inc.

# The AI control plane has five jobs

The questions enterprise buyers ask before scaling agents




© GitLab Inc.

# A governed flow compresses work, not accountability

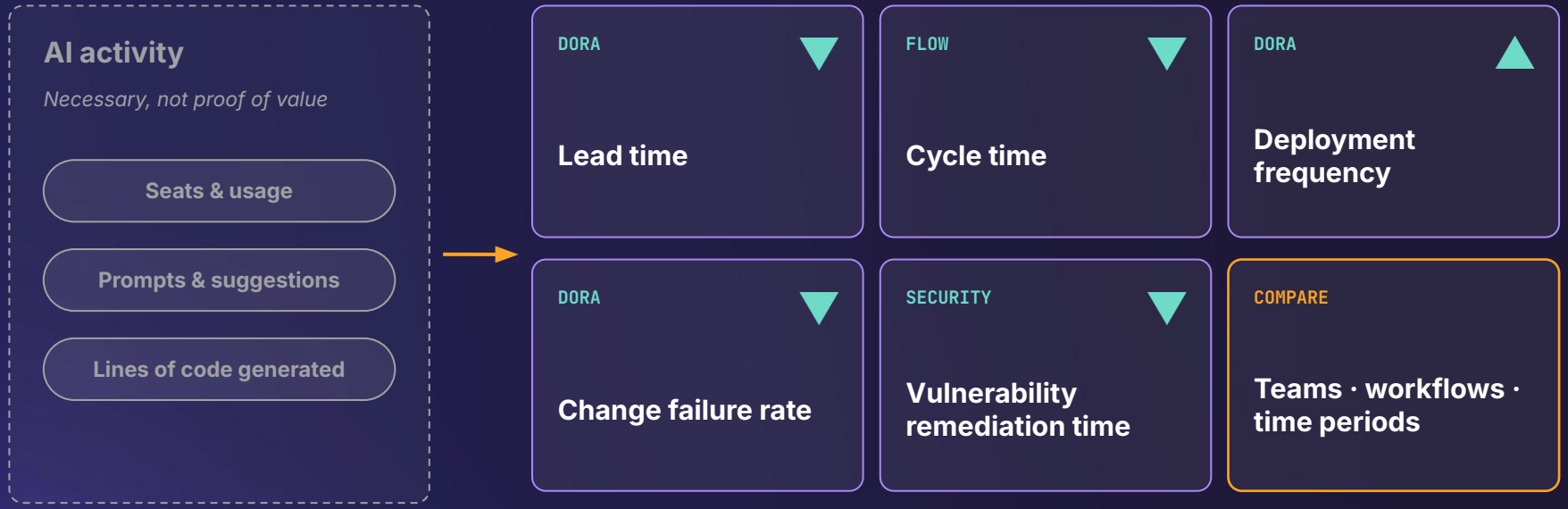
Agents execute the loop; humans keep decision rights at the control points




© GitLab Inc.

# Measure outcomes, not AI activity

Usage counts alone do not prove business value




Goal: measurable gains in speed, efficiency, and risk, not simply more generated code.

© GitLab Inc.

# Start with high-value bottlenecks

Begin where context, repetition, and handoffs create measurable delay

### Strong starting points

- Code review
- Failed pipeline resolution
- Vulnerability remediation
- Work item → merge request

Establish governance and measurement before expanding autonomy.




© GitLab Inc.

GitLab

# Customer proof

From AI adoption to delivery outcomes

**Barclays · Ericsson · Sigma Defense**



**AI adoption became an enterprise platform motion, not an isolated developer experiment.**

**We've been rolling out GitLab Duo to accelerate software delivery.**

**And it's not just about writing new code: we're refactoring legacy code, generating tests and detecting vulnerabilities.**

**Our developers are experiencing meaningful productivity gains – freeing them up to focus on higher-value problem solving.**

**Craig Bright**
CIO, Barclays

BARCLAYS

**48M+**
customers served

**18K+**
users for GitLab Duo

**>80%**
developer satisfaction

© GitLab Inc.

*Source: GitLab Barclays customer story and internal customer presentation materials.*

BARCLAYS

# AI adoption curve in one year

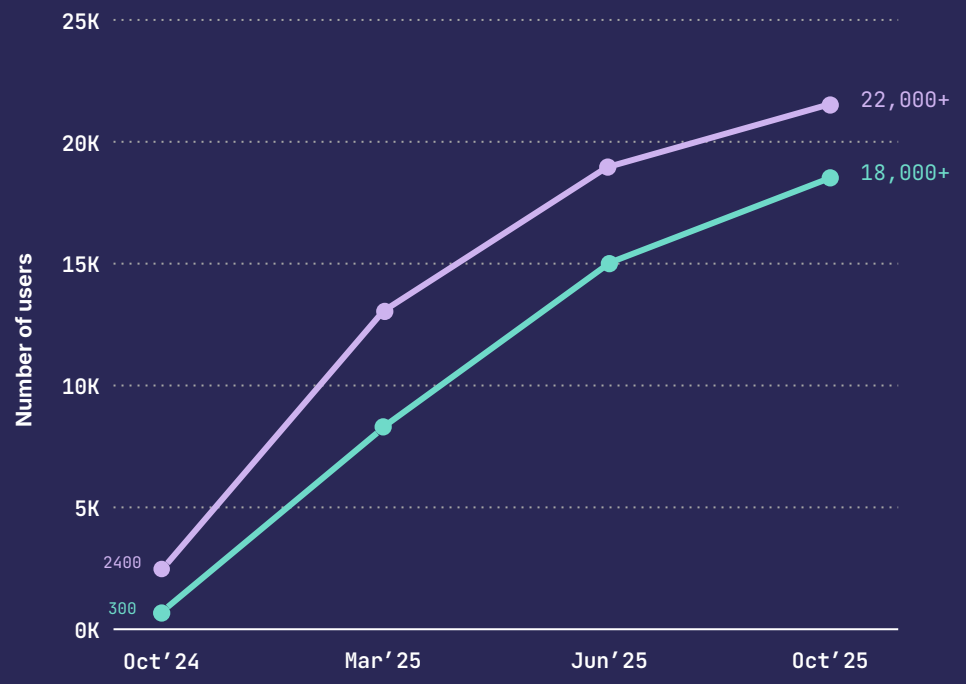
GitLab users

GitLab Duo users

**9x** growth

**60x** growth

Duo adoption grew 60× while GitLab user adoption grew 9× in the cited period.




© GitLab Inc.

*Source: GitLab Barclays customer story and internal customer presentation materials.*

# Ericsson: compressing time to market

Unified delivery workflows turned platform context into measurable delivery speed.

BEFORE



Upgrade — Multi-year wait for the next upgrade cycle

AFTER



New features and security patches every month

**Connecting delivery, testing, and operations, not only code generation.**

*"We can launch new features and security patches every month instead of waiting three years between upgrade cycles." Daniel Costa Soares, Ericsson OSS/BSS*

**50%**

faster deployments

**130,000**

hours saved over six months

**10×**

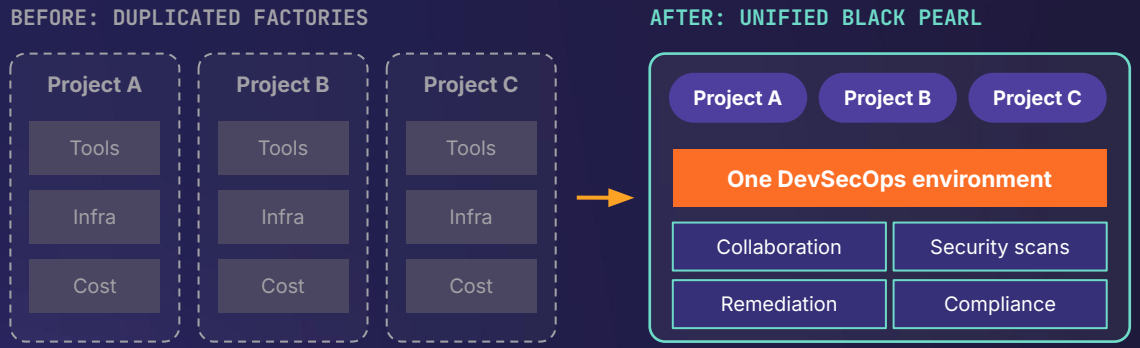
increase in testing scenarios

© GitLab Inc.

Source: GitLab Ericsson customer case study, https://about.gitlab.com/customers/ericsson/

# Sigma Defense: speed with security

Governed platform consolidation improved both mission speed and operational economics.




**Critical vulnerabilities could be addressed in hours rather than years.**

Less duplication; centralized collaboration, scanning, remediation, and compliance.

**98%**

reduction in onboarding time

**97%**

reduction in time to fix bugs

**90%**

reduction in software factory costs

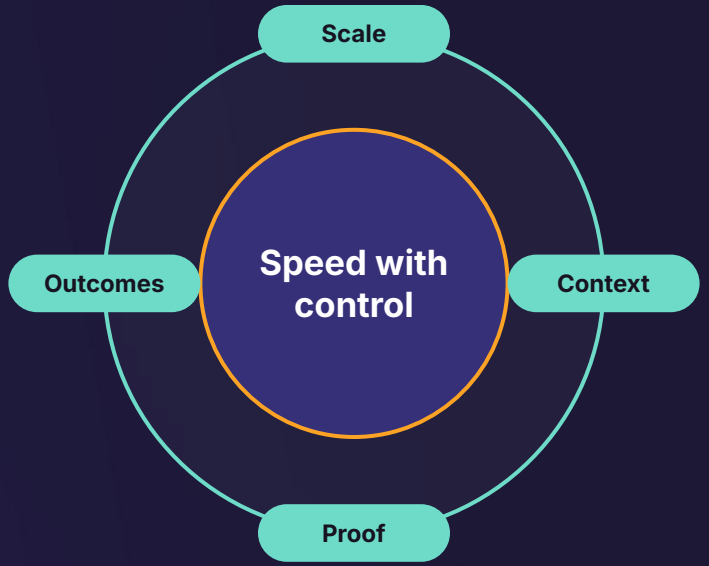
© GitLab Inc.

*Source: GitLab Sigma Defense customer case study, https://about.gitlab.com/customers/sigma-defense/*

# The new competitive question

~~Not "Which AI coding tool do you use?"~~

1. SCALE
   Can your software delivery system operate at agent scale?
2. CONTEXT
   Can agents understand the full lifecycle, not just the repository?
3. PROOF
   Can security and compliance prove what happened?
4. OUTCOMES
   Can leaders measure whether AI improved delivery outcomes?




© GitLab Inc.

# The future is not humans versus agents. It is teams conducting agents with speed and control.



© GitLab Inc.