UNIVERSITY OF MINHO · JUNE 2026

Agentic AI in the Software Development Lifecycle

Real-World Experiences:
AI Coders, DevOps and Proprietary Tools

Ernesto Pedrosa · Founder & CPTO, Automaise

automaise

AGENDA

automaise

01	**Setting the Stage** How LLMs changed software engineering — and where Automaise fits	3 min
02	**AI-Born vs. Legacy Companies** Why speed of adoption differs — and what that means for engineering teams	3 min
03	**Code Generation Tools** Hands-on with GitHub Copilot & Cursor: wins, fails, workflow shifts	7 min
04	**LLMs in DevOps / CI-CD** AI-assisted testing, code review, anomaly detection in pipelines	7 min
05	**Building Our Own Tools** Why we went custom — agents, automation & artefact generation	8 min
06	**Lessons Learned** What actually changed, real risks, and the discipline AI demands	5 min
07	**The New Developer Role** Less syntax, more architecture — what the AI era demands from engineers	3 min

01 · SETTING THE STAGE

automaise

65%

of developers

use AI coding tools weekly (GitHub Survey 2025)

46%

productivity gain

reported on well-scoped coding tasks

∞

new complexity

prompt engineering, hallucinations, context limits

Automaise builds AI-powered productivity software for customer support. We live at the intersection of automation, LLMs and the messy reality of enterprise adoption.
Our engineering team has been using — and building — AI tools for years. Here's what we learned.

02 · AI-BORN VS. LEGACY COMPANIES

automaise

Why they move at different speeds — and what it means for engineering teams

AI-BORN		LEGACY
Stack	Designed around AI from day one	20+ years of systems to drag along
Data	Structured for model consumption	Siloed, messy, politically guarded
Team	AI-first mindset is the default	AI adoption met with resistance
Process	Workflows built with AI in the loop	AI bolted on at the edges
Debt	None — greenfield advantage	Massive — technical and cultural
Speed	Ship fast, iterate with AI, no politics	Isolated pilots, slow rollout, approvals

VS

The race: can incumbents transform before challengers scale? Legacy has the data & domain. AI-born has the speed & architecture.

03 · CODE GENERATION TOOLS

automaise

✓ What Works Well

- Boilerplate & repetitive code (models, DTOs, CRUD)
- Unit test scaffolding from existing functions
- Docstring & comment generation
- SQL query drafting from natural language
- Refactoring suggestions in well-scoped context

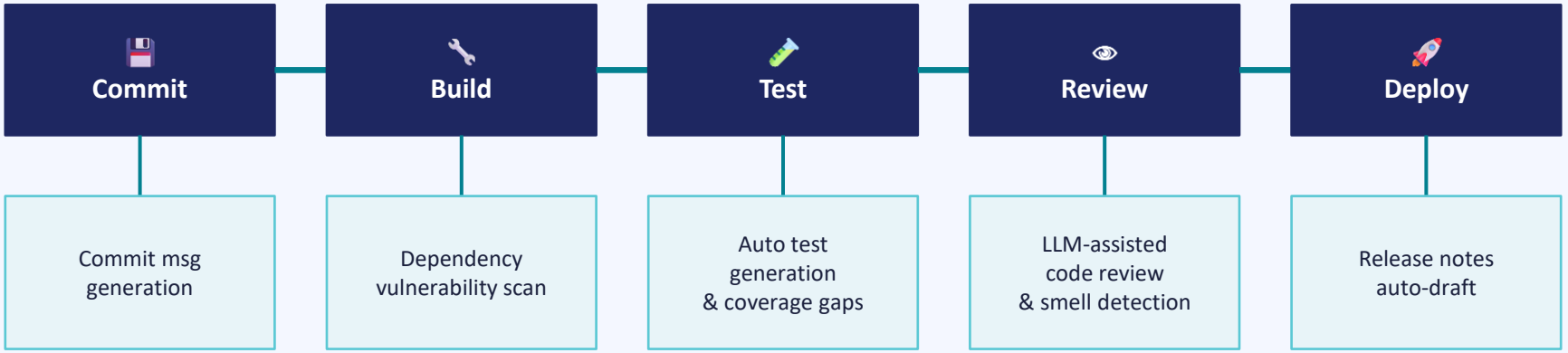
✗ Where It Fails

- Complex business logic with deep domain context
- Cross-service consistency & architecture decisions
- Security-sensitive code (trust but verify!)
- Hallucinated APIs / outdated library usage
- Long context degradation across large files

Key insight: Copilot & Cursor are force-multipliers for experienced engineers — not replacements.
Workflow shift: think in prompts, review critically, own every line committed.

04 · LLMs IN DEVOPS / CI-CD

automaise



Automated Test Gen

LLM analyzes diff and proposes unit tests for changed functions. PR gate requires coverage threshold.

Anomaly Detection

Embeddings on log streams flag unusual patterns before they become incidents. Early warning in prod.

Human Oversight

Every LLM suggestion is a PR comment, not an auto-merge. Engineers stay in the loop — always.

05 · BUILDING OUR OWN TOOLS

automaise

Why Go Custom?

- Off-the-shelf tools don't know our domain
- Data privacy & customer data constraints
- Tighter integration with internal systems
- Cost control at scale
- Competitive differentiation

AI Agents

Autonomous agents handle multi-step support workflows — triage, escalation, resolution — with human handoff.

Process Automation

LLM-orchestrated pipelines replace brittle rule-based systems. Dynamic, context-aware, explainable.

Artefact Generation

Conversation logs → structured summaries, reports, CRM entries. Saving hours per agent per day.

Architecture: RAG + fine-tuned models + tool-use agents. Evaluated on real production data. Continuously monitored.

06 · LESSONS LEARNED

automaise

📈 SDLC is faster — but also more complex

Iteration cycles shrink dramatically. But prompt engineering, eval frameworks, and LLM ops add new overhead.

⚠️ Risks are real

Hallucinations & “spaghetti” source code in production code, vendor lock-in, context window limits, and the subtle erosion of deep thinking.

🎯 Quality requires new discipline

LLM output demands systematic review. We test AI suggestions like we test code: with criteria, edge cases and CI gates.

👷 The engineer's role has evolved

Less syntax, more systems thinking. The best engineers use AI as a fast intern — they stay firmly in charge of architecture.

07 · THE NEW DEVELOPER ROLE

automaise

AI makes bad engineers worse.
And great engineers unstoppable.

LESS OF THIS

- Writing boilerplate by hand
- Googling syntax & API signatures
- Spending hours on repetitive refactors
- Being blocked by unfamiliar languages

MORE OF THIS

- Systems thinking & architecture
- Prompt design, evals & output validation
- Security, ethics & trust boundaries
- Knowing when NOT to use AI

The floor has risen. The ceiling has too. The question is: where are you building?

automaise

Demo time

A Real-world example – Max, The Coder.

Thank you.

Questions & Discussion

Ernesto Pedrosa
Founder & CPTO · Automaise
ernesto@automaise.com

automaise.com

automaise