"There's always a new frontier.
The only question is:
will you cross it?"

MOTIVATION

Why leave your comfort zone?

The quality of local opportunities didn't match my ambitions

INE ESTIMATE

Where do Asturian graduates end up?

- **63.3%** Work in Asturias
- **26.6%** Other Spanish region
- **10.1%** Work abroad

36.7% of graduates leave Asturias

"Asturias isn't losing
talent...
it's **funding global talent**."

GENEVA, SWITZERLAND

What I learned at CERN

- How to be **independent** — figure things out on your own
- How to manage **continuous learning** — it can be very demanding
- How to **ask for forgiveness, not permission**
- How private cloud infrastructure actually works (OpenStack)

FROM EXPERIENCE

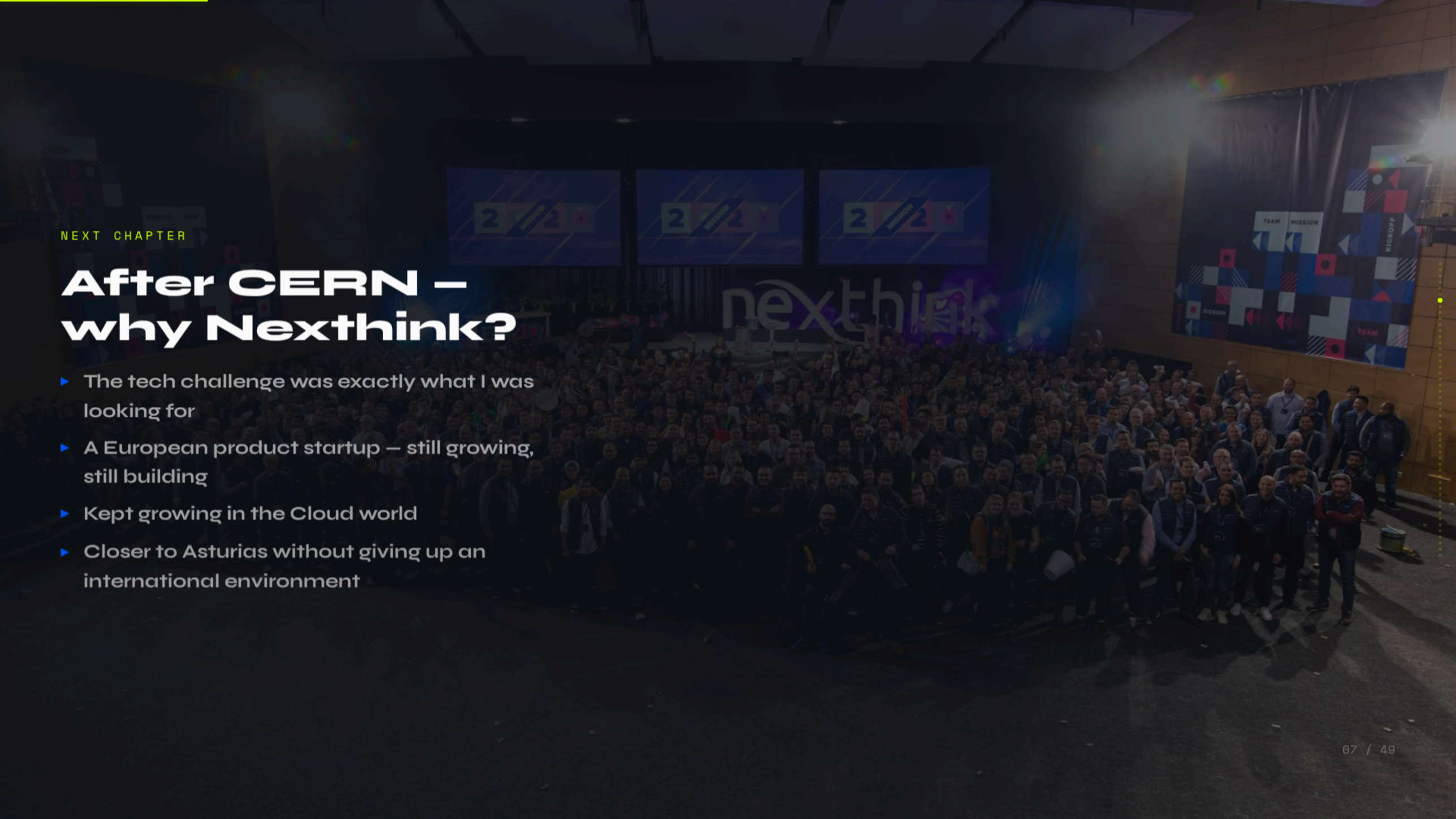
My advice for students

- **Take risks.** You're at the perfect moment in life to do it.
- **Go abroad** — not just within Spain. The experience compounds.
- **Start a side project** right now — you'll learn more than any course.
- **Come back** and help make Asturias better.

NEXT CHAPTER

After CERN — why Nexthink?

- The tech challenge was exactly what I was looking for
- A European product startup — still growing, still building
- Kept growing in the Cloud world
- Closer to Asturias without giving up an international environment



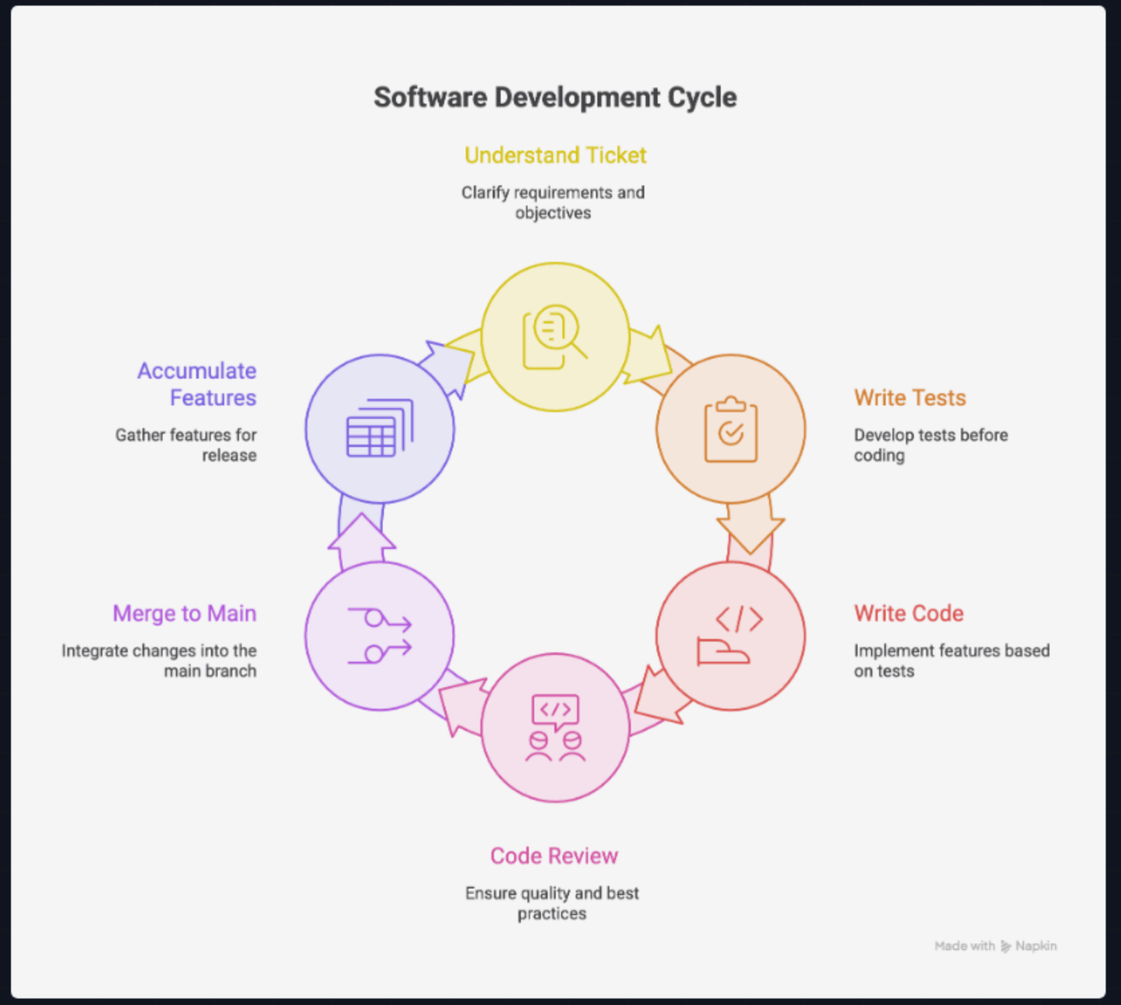
ACT I

Before Cloud Native

How software was built and shipped
— before everything changed

THE OLD WAY

The old development cycle



ARCHITECTURE

The old architecture & environment

- Monolithic architecture
- Installed on customer VMs (CentOS 6/7)
- " **Snowflake** " test environments — unique, fragile, not reproducible
- Dedicated QA profiles within each team

THE CONSEQUENCES

The limitations this created

- Tight coupling between all installed components
- Hard to maintain: multiple versions, slow updates
- Features took **months** to reach customers
- Infrastructure treated like a **pet**, not cattle

"It took us 6 months
to install the product
at one customer."

ACT II

Cloud Native

A different way to build, deploy, and operate software

DEFINITION

What does Cloud Native mean?

- Software designed to **run in the cloud** from the start
- The **"cattle vs. pets"** mindset shift:
  - Pets: named servers you nurse when sick
  - Cattle: identical, replaceable, disposable
- Infrastructure is **code**, not configuration

ORGANIZATIONAL DESIGN

Conway's Law

> *"Any organization that designs a system will produce a design whose structure is a copy of the organization's communication structure."*
>
> — Melvin Conway

- Big, monolithic team → monolith
- Small, autonomous teams → microservices

ARCHITECTURE PATTERNS

Cloud Native architectures (1/2)

- **Microservices** — small, independent, deployable services
- **Serverless** — no servers to manage; pay per invocation
- Each service owns its own data and lifecycle

ARCHITECTURE PATTERNS

Cloud Native architectures (2/2)

- **Event-driven** — services communicate via events, not direct calls
- **Circuit Breaker** — stop cascading failures automatically
- Resilience and isolation are **built in by design**

STACK OVERVIEW

A typical cloud product stack

- Cloud provider (AWS, GCP, Azure)
- DNS → API Gateway → Routing
- **Kubernetes** for container orchestration
- **Managed databases** (no more DBA ops)
- Observability stack (logs, metrics, traces)

CONTAINER ORCHESTRATION

Kubernetes — what it is & why it matters

- Runs your containers **at scale**
- **Autoscaling:** adds/removes pods based on load
- Handles failures automatically (self-healing)
- Developer impact: write a manifest, K8s does the rest

EVENT STREAMING

Kafka — event streaming at scale

- **Pub/Sub mode:** producers publish, consumers subscribe
- **Stream mode:** process events as they flow (real-time analytics)
- Use cases: audit logs, real-time pipelines, decoupled microservices

STARTUP PERSPECTIVE

Do you need all this in a startup?

- **No. Start small.** Don't over-engineer.
- But: organize your monolith so splitting it later is "cut and paste"
- Separate domains clearly from day one
- The architecture will follow your team structure (Conway again)

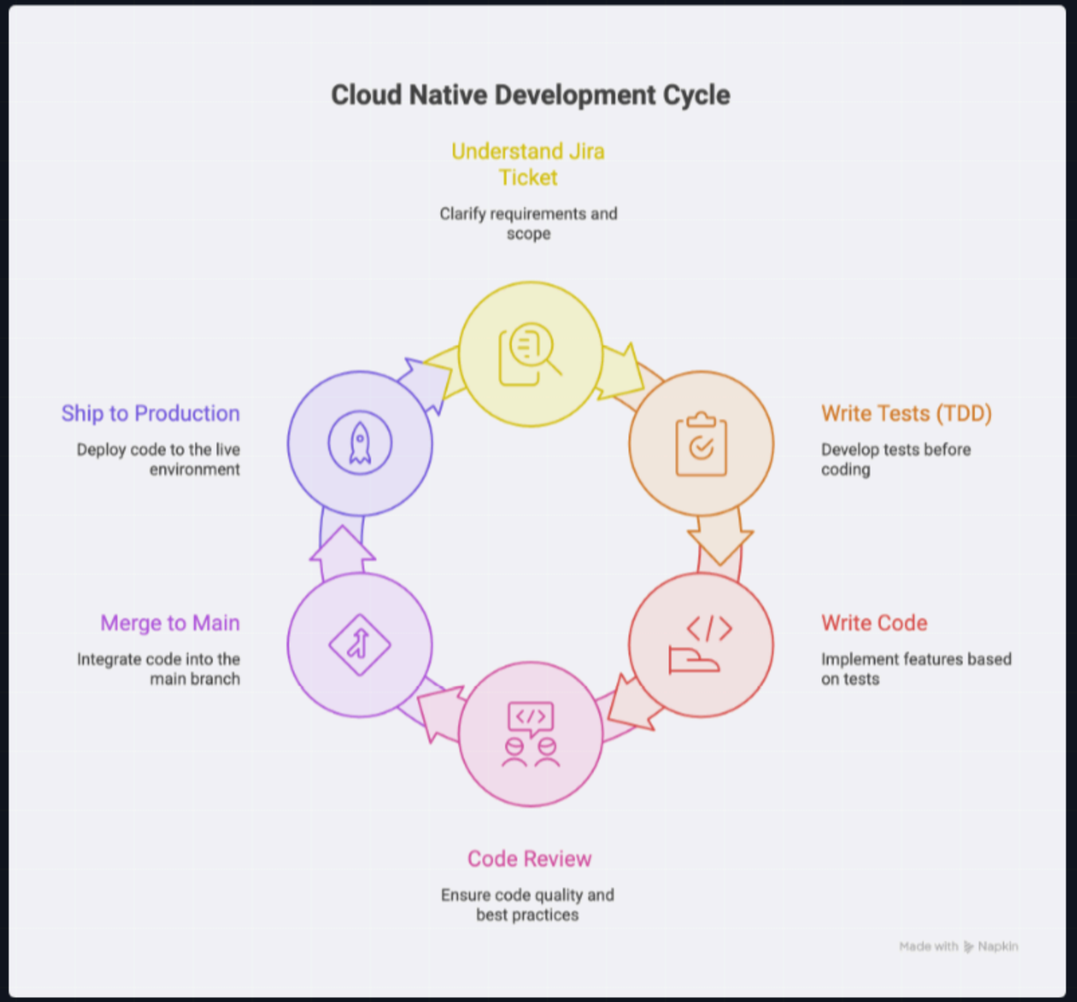
"The worst outcome: a microservices architecture that's still a monolith at heart."

WORKING STYLE

How to work in a Cloud Native company

DEVELOPMENT PROCESS

The Cloud Native dev cycle



THE TWO PILLARS

Two pillars of Cloud Native development

- **CI/CD** — fast, automated path from code to production. A bug can be fixed in production in *15 minutes* .
- **Observability** — logs, metrics, traces. If you can't see it, you can't fix it.
- QA is no longer a separate role — **every engineer owns quality**

BEFORE → NOW

Limitations — resolved

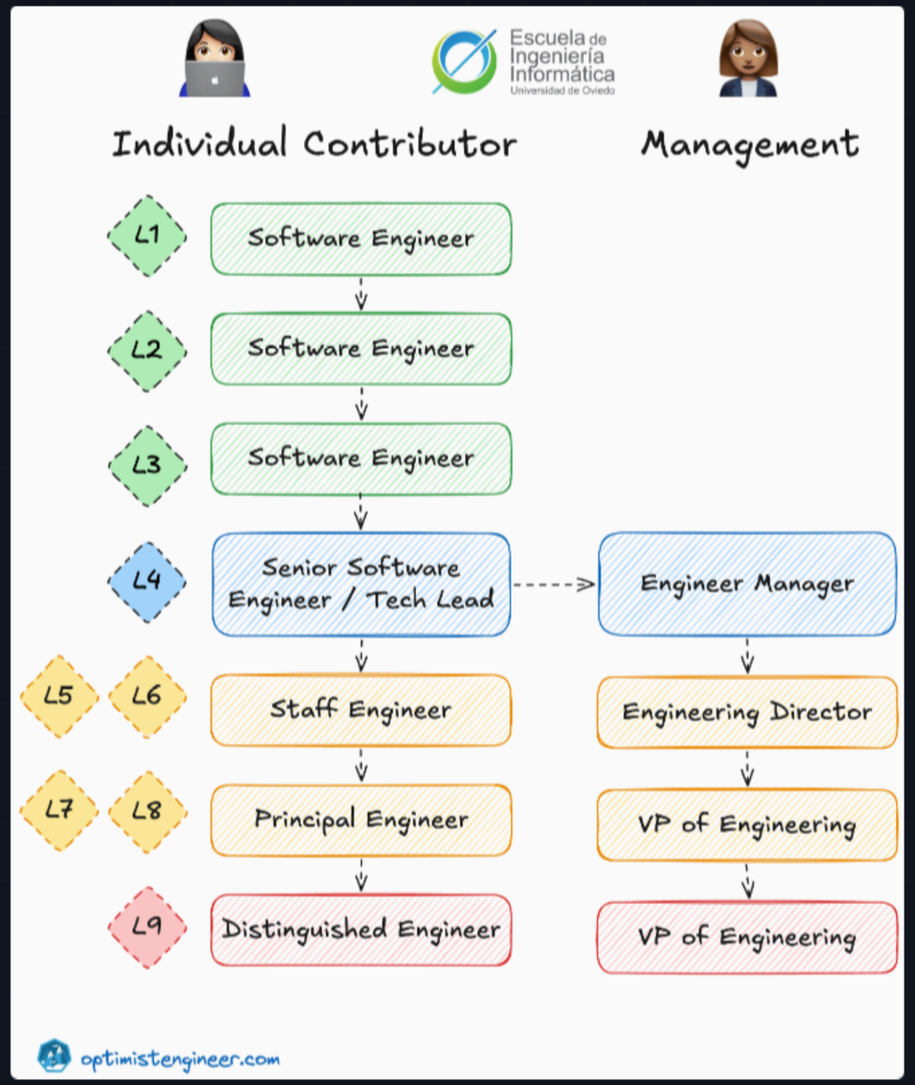
- Tight coupling → everything decoupled by design
- Slow updates → continuous deployment, always up to date
- Slow feature delivery → features can reach customers in minutes

CAREER GROWTH

How to grow professionally

COMPANY STRUCTURE

Typical org chart


Escuela de Ingeniería Informática
Universidad de Oviedo
Individual Contributor
Management
L1
Software Engineer
L2
Software Engineer
L3
Software Engineer
L4
Senior Software Engineer / Tech Lead
Engineer Manager
L5
L6
Staff Engineer
Engineering Director
L7
L8
Principal Engineer
VP of Engineering
L9
Distinguished Engineer
VP of Engineering
optimistengineer.com

IC TRACK

Individual Contributor levels

- **L1–L3:** Learning, executing tasks, building foundations
- **L4:** Owns features end to end. Makes good technical decisions independently.
- **L5–L6:** Leads projects. Raises the bar for the team.
- **L7–L8:** Staff / Principal. Shapes technical direction across teams.
- **L9:** Distinguished. Company-wide technical impact.

MANAGEMENT TRACK

The Manager path

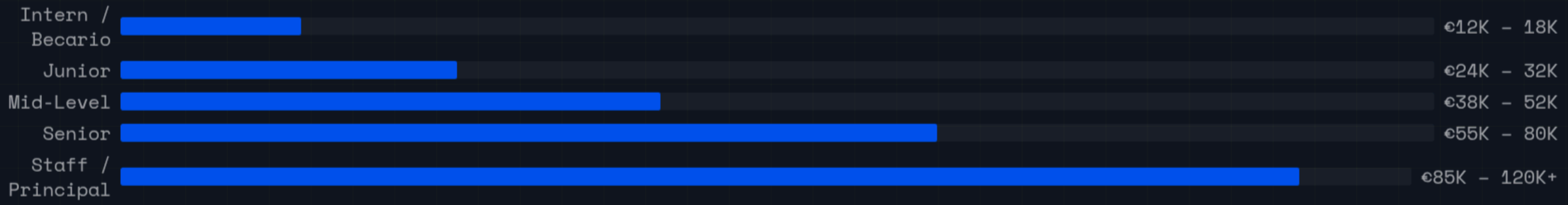
- At a certain seniority level (typically L5+), you can move to **Engineering Manager**
- Managers lead **people, not code**
- The IC and Manager tracks are equally respected in mature product companies
- Neither path is "better" — choose based on **what energizes you**

"Act in the role
before you get the role."

"Make your
work visible."

SPAIN, 2026

Salaries by experience level



Source: INE × Tecnoempleo × Infojobs, 2026

SPAIN, 2026

Salaries by region

Source: INE × Tecnoempleo × Infojobs, 2026

EUROSTAT, 2026

IT salaries across Europe

Source: Eurostat 2026, NACE J62 – Programming & Consultancy

ACT III

AI Native

The next shift is already here

DEFINITION

What does AI Native mean?

- Cloud Native changed *how we deploy* software
- AI Native changes *how we create* software

> *"AI is a* **first-class collaborator** *at every step of the dev lifecycle — from reading the ticket to writing, testing, and reviewing code. The human shifts from* **writer to director**.*"*

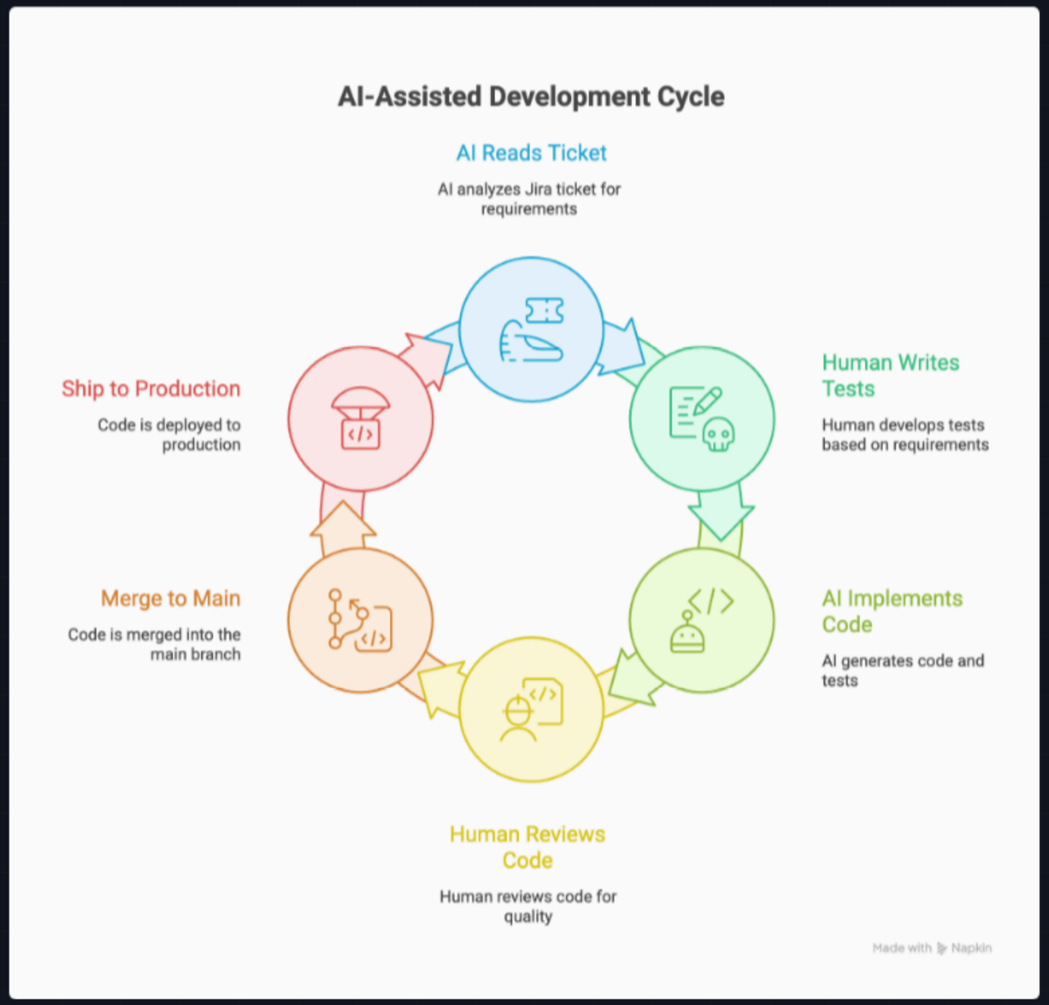
IN PRACTICE

How companies adopt AI tooling

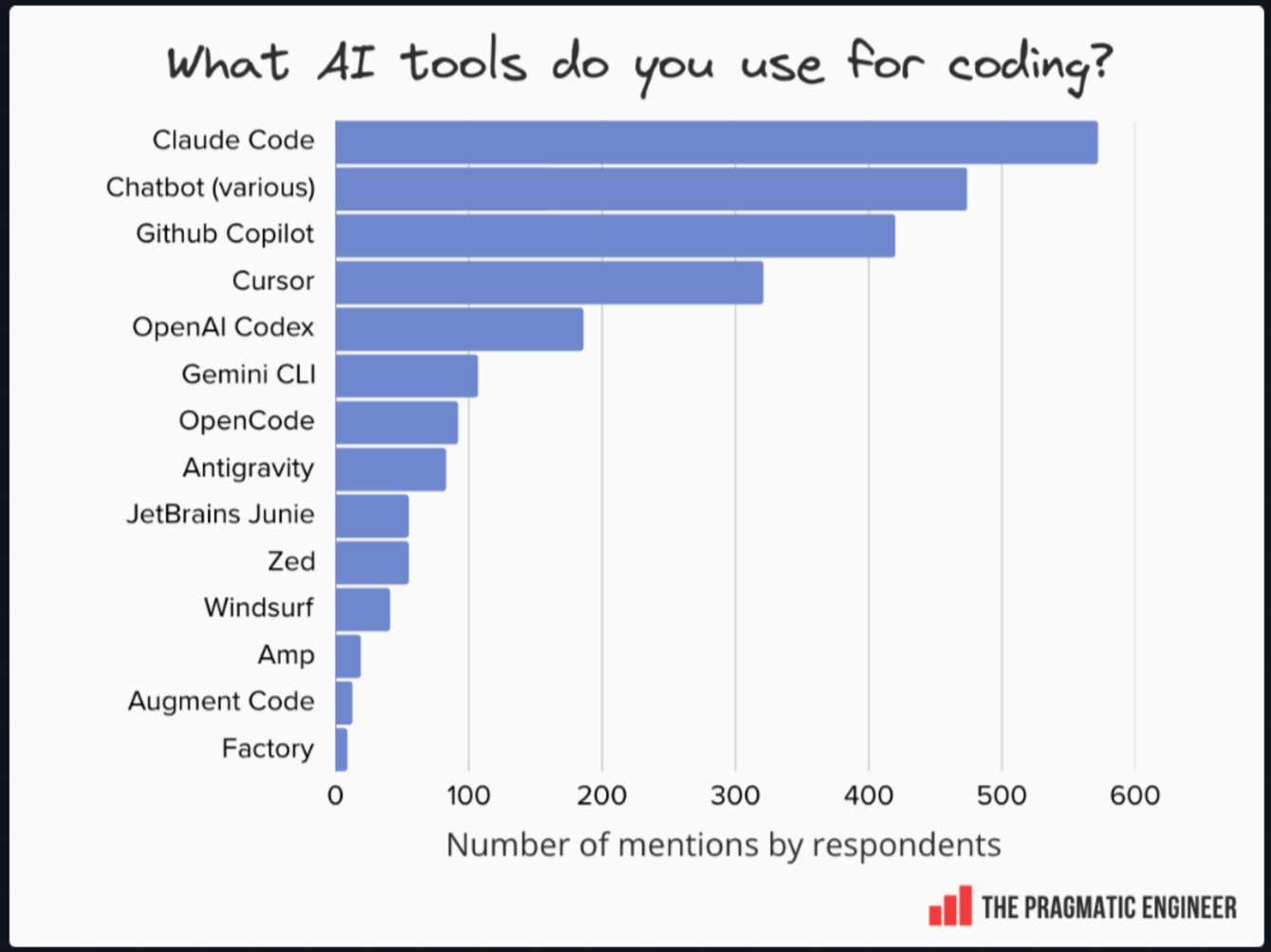
- The relevant team analyses the market
- Developers are asked what they prefer
- Licenses and training are purchased (basic + power users)
- In the case of AI: **this cycle repeats constantly**
- Example: Cursor suddenly hiking prices for enterprise customers

NEW WORKFLOW

The AI-assisted dev cycle



AI coding tools — industry snapshot (2026)



Source: The Pragmatic Engineer, 2026

THE PROBLEM

The Code Review Problem

- AI generates more code than humans can review line by line
- Code review becomes a **bottleneck** — or worse, **theater**
- The reviewer approves without truly understanding what changed
- Trust erodes. Quality degrades. Nobody notices until production.

THE SOLUTION

Solution — Human in the Loop

- AI generates an **ADR** (Architecture Decision Record) *alongside* the code
- The human reviews **the ADR**, not the code line by line
- ADR captures: what was decided, why, and what alternatives were rejected
- The human evaluates the design decision, not the mechanical implementation

LIVE DEMO

AI-assisted coding with Cursor

AI writes the code

→

AI generates the ADR

→

Human reviews the ADR

THE BIG QUESTION

Will AI take our jobs?

- Companies want you to use AI as much as possible
- They don't yet know if it makes more money — but it attracts investors
- Mid-level engineers are scared. That's understandable.
- Measuring productivity with AI is still an unsolved problem

THE ANSWER

The adapters survive

- This is **a new layer of abstraction** — not the end
- Punch cards → assembly → high-level languages → **AI**
- Those who adapt are the ones who thrive. That's always been true.

STATE OF THE ART

Agents & subagents

- We are entering the **agentic era**
- **Agents:** autonomous AI that can plan and execute multi-step tasks
- **Subagents:** specialized agents called by a coordinator
- Reference framework: **AGENTS.md** — defines agent behavior and boundaries
- My biggest concern: **Engineering Manager and Product Manager roles**

ONE LESSON PER CHAPTER

What I'd tell my past self

- **Dare to leave your comfort zone** — the best opportunities are rarely where you already are. The spirit matters more than the destination.
- **Learn the stack, not just the syntax** — Cloud Native is a mindset shift, not a tool list. Architecture, CI/CD, observability: this is the craft.
- **Act in the role before you get the role** — ownership and visibility compound over time. IC or manager — both paths lead somewhere real.
- **Adopt AI, don't fear it** — this is just another layer of abstraction. Those who adapt are those who thrive. That's always been true.

"AI is the new frontier.
The only question is: are
you going?"