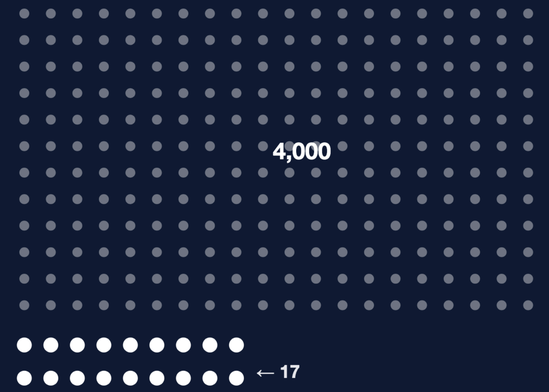
17 People.
4,000 Engineers.
Zero Authority.

How a startup changed a legacy organisation from the inside

JEEVAN D C · SSH LIVE 2026



What we were told on Day 1

"Most of them wouldn't want us there."

We entered an environment designed to resist change

The setup (illustrative)

- 17-person startup, acquired into a **4,000+ person** organisation
- 30-year-old technology estate
- New city, new hub, **no existing relationships**
- Leadership mandate: one production change per month

> **We needed an S3 bucket. Writing the IaC — 30 minutes.**
> **Getting permissions through process — 13 days.**

We judged the system before understanding it — and they judged us back

The instinct vs. the reality

Our instinct

"Why is the architecture like this?"

"Why no test cases?"

"Why is deployment manual?"

Given their timelines and context —
the decisions made sense.

What the org saw

- Outsiders. A threat to their autonomy.
- Wouldn't give us **read access**
- Territorial about their systems

We believed our way was the right way.
That's its own arrogance.

We couldn't change them — so we changed what we could, visibly

Locus of control (illustrative)

COULDN'T CONTROL

- Everything manual
- Teams wouldn't let us in
- No authority to change anything
- Leaders frustrated

COULD CONTROL

- Our own team's practices
- RFCs, ADRs, documentation
- Test cases from day zero
- Automation on all new work

Did it visibly. Waited.

Our first platform engineering attempt died in 3 weeks

Backstage Internal Developer Portal — failure case

Couldn't get a VM
Infra team backlog

Process overhead
Approvals killed momentum

Died as POC
Zero contributions after week 3

Repository activity

0 contributions in the last year

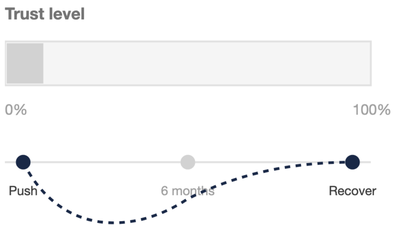
The org wasn't slow because people were slow — it was optimised for the appearance of control, not the reality of it

Remember this slide

We damaged a relationship for 6 months by being right

Trust recovery timeline

- Pushed too hard on one team's deployment process
- It got escalated
- Relationship damaged for **six months**



> **Being right ≠ right to force the solution.**
> **Some resistance was legitimate governance.**
> **Our confidence was built on survivorship bias — not universal truth.**

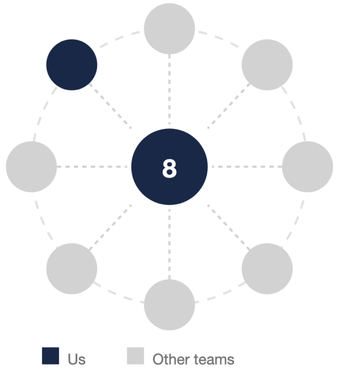
The turning point — a budget problem created the collaboration the org never had

Voluntary collaboration across 8 teams

- No budget to hire translators for Portuguese-based regions
- Top-down side project from the CPTO
- **8 teams** came together — voluntarily
- No Agile lead. No scrum ceremonies.

Result:

RAG system — 16 languages, 25 countries.
First-ever IaC in the org. First true cross-team collaboration.



"We're fine. It's them."

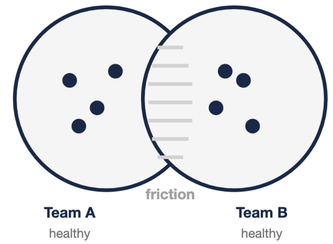
The Insular Health Fallacy

Every team looked healthy by local measures:

- Velocity stable, retros positive
- Psych safety high — people admit mistakes
- Good code, good practices, good intent

But at every boundary:

- 3-week waits for a PR review from another squad
- APIs breaking without notice
- Planning sessions that felt like hostage negotiations



> **Internal health ≠ system health**
>
> **A team can be locally excellent and still generate friction at every boundary**

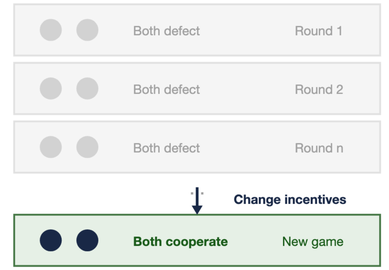
Defection was the dominant strategy — the game was broken, not the people

Payoff matrix (Prisoner's Dilemma)

	B COOPERATES	B DEFECTS
A COOPERATES	**Win–Win** Both ship faster	A loses B ships on time
A DEFECTS	A ships on time B is blocked	**Lose–Lose** Zero trust ← default

> Every team making the smartest choice for themselves — and the whole system grinding to a halt. The Price of Anarchy

REPEATED INTERACTIONS OVER TIME



Cooperation becomes the dominant strategy

when payoffs are restructured

"How can we help you so you can help us?"

Changing the game — mechanisms that shifted the payoff structure

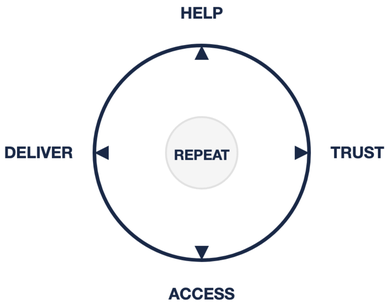
→ Offered to automate their manual work. No strings attached.

→ Wrote test cases and docs for their codebases.

→ Proposed shared OKRs: your success depends on them.

→ Cross-team reviews tied to appraisals. Public appreciation.

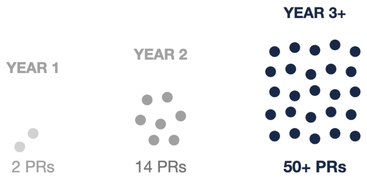
> **No single thing worked alone. The combination moved the needle.**



3.5 years from "no access" to cross-pillar contributions

InnerSource adoption timeline

MONTH 0	MONTH 4–14	MONTH 24	MONTH 36+
No access	Read access → Docs → Tests	Code → Production	**Cross-pillar**



3.5 years. One PR at a time.

Some teams never opened up. We focused on the willing.

You can't fix what people don't see as broken

Normalised pain

WHAT YOU SAY

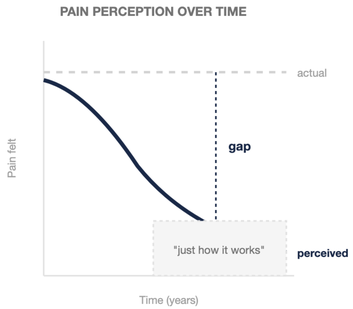
"Let me help fix this."

WHAT THEY HEAR

"You've been doing it wrong."

When people live with a broken process for years, **they stop seeing it as broken.**

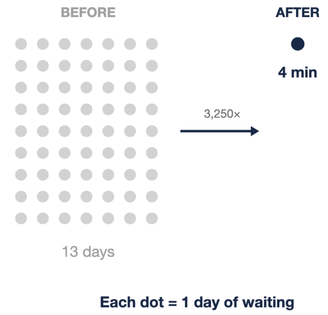
> **Wait for them to feel the pain themselves. Then be ready with the help.**



The delta that proved the approach works

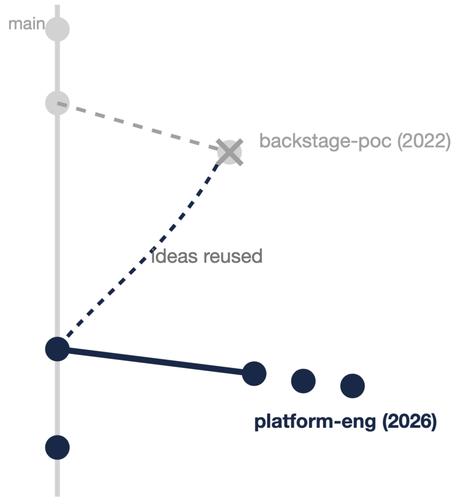
Key metrics — before and after (illustrative)

Metric	Before	After
Environments	Excel sheet, 13-day wait	**Self-service, 4 min**
Deploy frequency	Every 2 months	**Every 10 days**
Squads adopting	0	**25+ (RFCs, ADRs, tests)**
Hub size	17	**100 → influencing 4,000+**



The POC that died became the platform that lives

Remember Backstage? — same outcome, different path



- **2022:** Couldn't get a VM. POC died.
- **2026:** Influenced platform engineering through InnerSource.
- **Today:** Anyone — PMs, QAs — can spin up a POC environment.

13 days → 4 minutes

Same outcome, different path

"*Seeds grow where the soil is ready — not where you plant them.*"

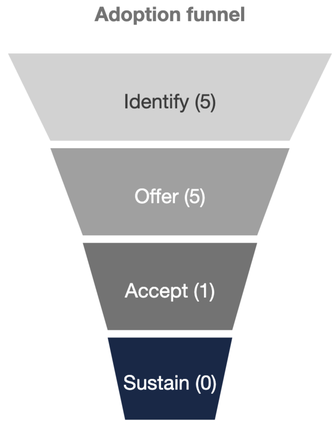
— Meadows, Thinking in Systems

5 initiatives that failed — and the pattern underneath

Failure inventory

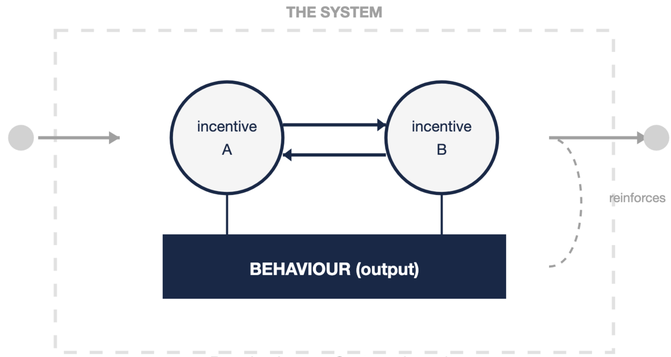
Initiative	Outcome
IDP (Backstage)	Couldn't get a VM. Died as POC
SRE practices	Offered to help. Teams got territorial
Kafka automation	Wanted to automate. No traction
Shared libraries	No adoption
Universal InnerSource	Some teams never opened up

> **The pattern:** We saw it → We offered → They said no → We had no authority.
> Not every "no" was wrong.



It's the system

Not the people. Not the culture. The incentive structure.



People change. System doesn't.

From blaming people to redesigning systems

How my thinking changed

YEAR 0	YEAR 1	YEAR 3
"People are the problem."	"It's their mindset."	**"It's the system."**

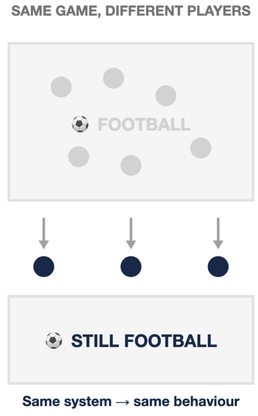
⚽ **The Football Team**

Swap every player one by one. New player sees football → plays football. **The system reproduces itself.**

"A bad system will beat a good person every time." — Deming

🏀 **What we hoped**

Swapping players would change the game. It won't.



How I enter any new problem today

The playbook

ACTIONS

1. "How can I help?" — not "what's broken"

2. Business problems first, tech last

3. Ask questions — let them arrive

4. Make them own it — enable, don't hero

5. Let them take the credit — you disappear

6. Let others win the battle to win the war

PRINCIPLES

7. Judge systems, not people

8. Win-win or no deal

9. Autonomy with guardrails

10. Play to strengths

> **"Trust compounds. Being right doesn't."**

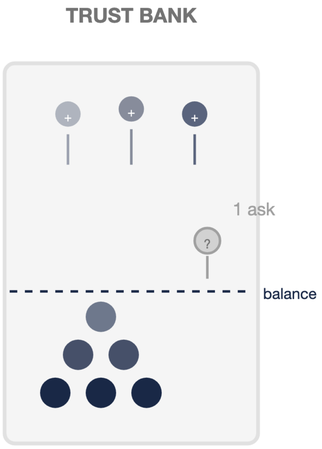
Three principles — each with a concrete mechanism

What worked for us

1 **Change the game, not the players**
Mechanism: Shared OKRs across dependent teams.

2 **Start with the willing — one team, one practice**
How to find them: Look for the team already frustrated.

3 **Make three deposits before any withdrawal**
Write their docs. Fix their tests. Build the balance. Then ask.



3 deposits

This isn't about team size — it's about willingness

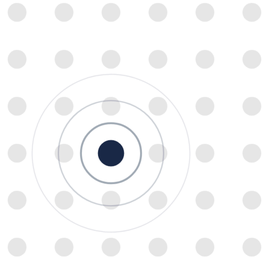
Transferability

~~You don't need 17 people.~~
~~You don't need to be an acquired startup.~~

You need one team willing to go first.

The signal:
They're already frustrated with the status quo

That might be you. That might be someone three levels below you.
Either way — find them.



One team. Ripple effect.

"Most of them wouldn't want us there."

Three and a half years later,

they don't want us to leave

Not because we fixed their systems.

Because we helped them see what they were capable of.

One of our engineers told me:

> *"My stagnant career changed. I'm confident now — even if I want to move."*

> Not the deploy frequency. Not the build times. **The person**

Questions?

📪 Get in touch: jeevan.dc24@alumni.iimb.ac.in

🌐 I write at https://noobj.me