We’ve all experienced the frustration of a broken customer journey.

Think about the last time you had to repeat your issue to three different support agents. Or when an online promotion wasn’t honored in-store. This is the “multichannel” problem: multiple channels exist, but they don’t talk to each other. The result is inconsistency, repetition, and a poor experience for the customer.

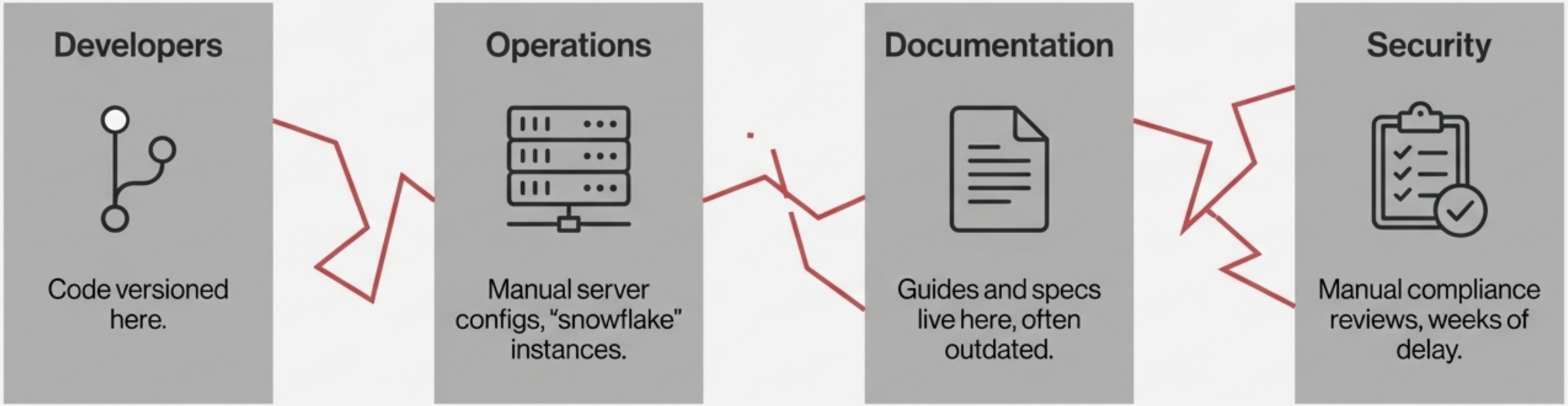
The goal is an ‘omnichannel’ experience, where every channel works together.

An omnichannel strategy integrates all touchpoints into a single, seamless journey. touchpoints into a single, seamless journey. The customer moves effortlessly between channels without losing context. losing context. Data is shared, the conversation is continuous, and the experience is consistent.

This is the modern standard for customer experience. According to McKinsey, 75% of consumers now expect it.

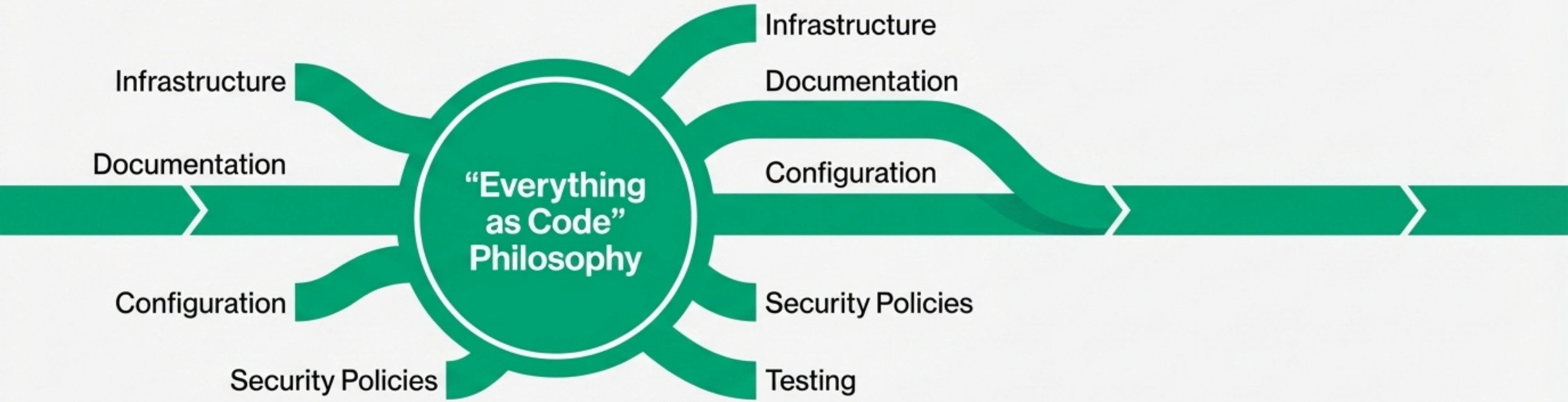
Our development lifecycle often suffers from the same “multichannel” problem.

Our teams and tools often operate in silos. Infrastructure is configured manually, documentation lives in a separate wiki, code sits in Git, and compliance is a final, disconnected step. Each “channel” has its own process and its own “truth,” leading to inconsistency, rework, and risk.



‘Everything as Code’ is the omnichannel blueprint for development.

“Everything as Code” is a philosophy that applies software engineering best practices—like version control, automation, and testing—to every part of the development lifecycle. It treats infrastructure, documentation, configuration, and governance not as separate manual tasks, but as code. This creates a unified, automated, and consistent process from start to finish.



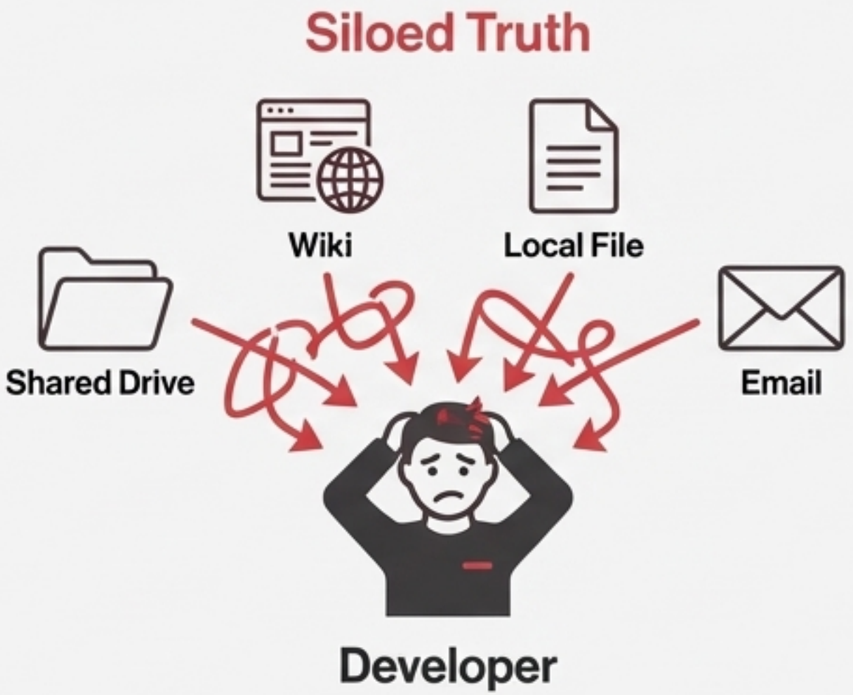
The foundation of this blueprint is a Single Source of Truth (SSoT).

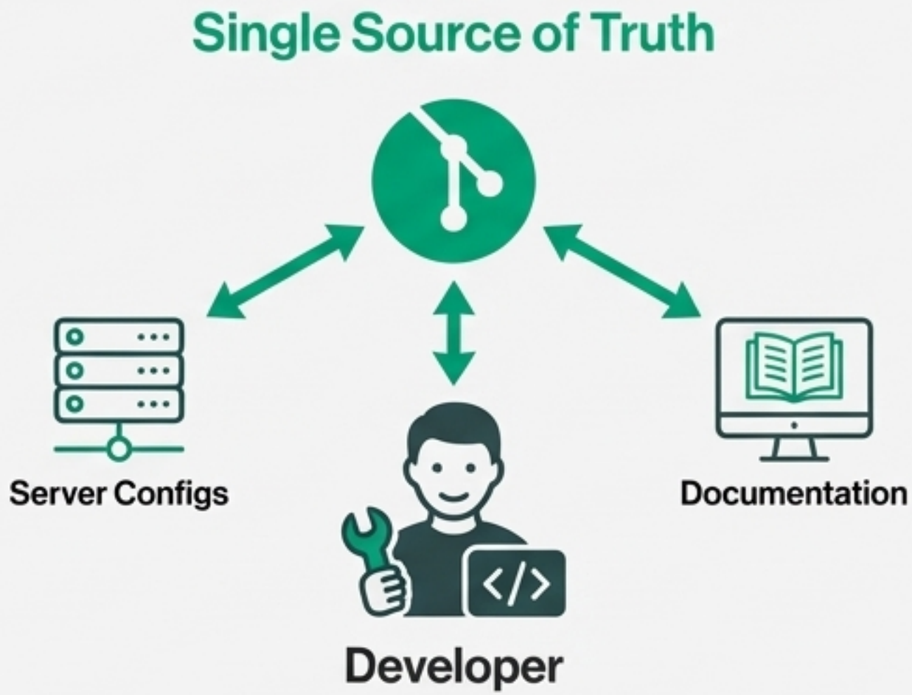
An SSoT is a centralized repository that guarantees everyone in an organization has access to the same, up-to-date information. It eliminates data silos, confusion, and rework.

In the "As Code" world, the version control system (like Git) becomes the SSoT for everything from application code to server configurations and user manuals.

Key Benefits

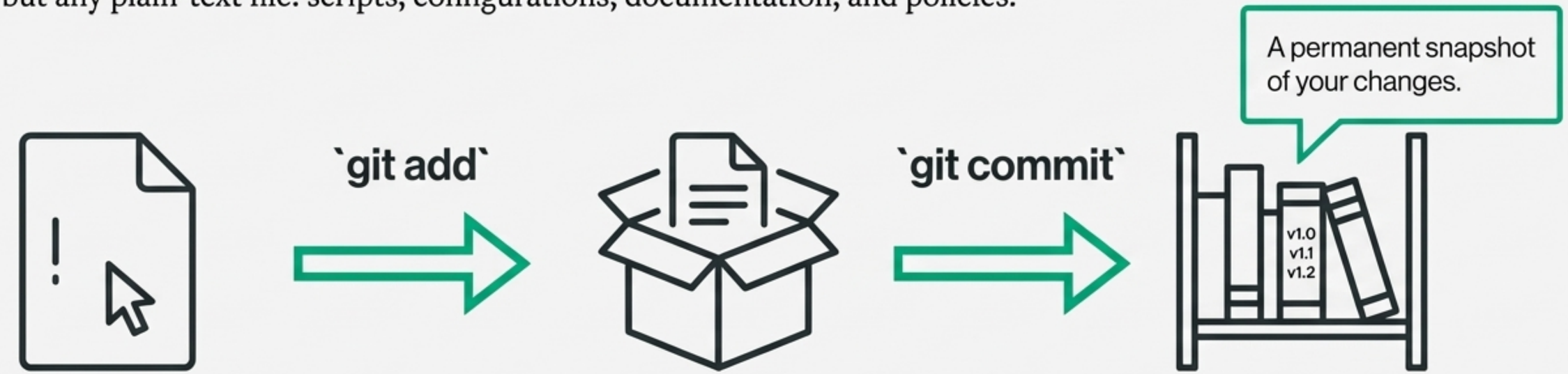
- Stops endless searching for the right file; employees spend ~29% of their week just searching for information.
- Avoids useless work based on outdated versions.
- Builds trust and consistency.





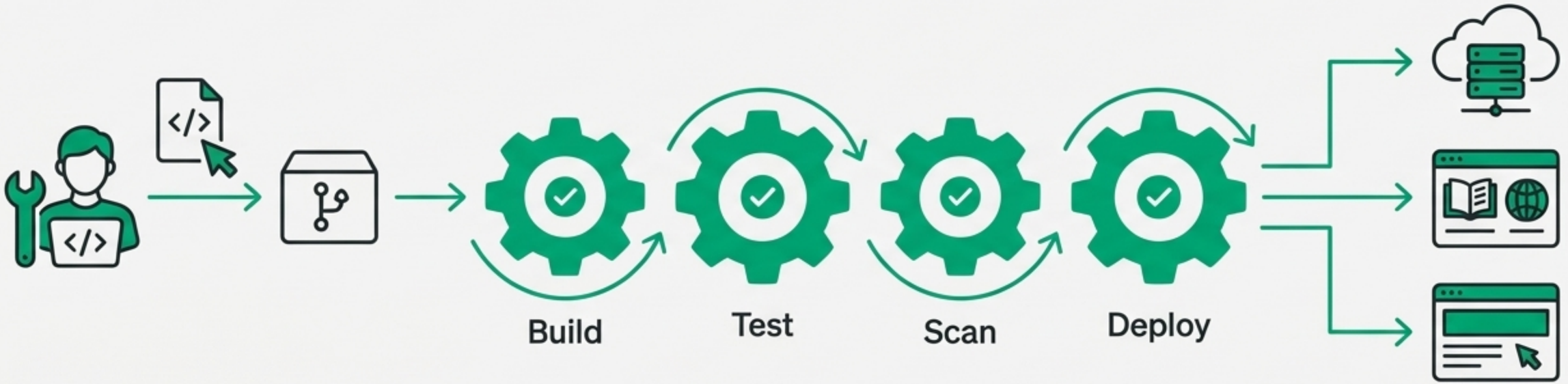
Version control is the mechanism for managing the SSoT.

A Version Control System (VCS) like Git tracks the iterative changes made to files. It's a 'save' button with a perfect memory, allowing you to record snapshots (commits) of your work, see a full history of changes, and revert to any past version. This is the key to managing not just source code, but any plain-text file: scripts, configurations, documentation, and policies.



Automation (CI/CD) is the engine that brings it all to life.

Continuous Integration and Continuous Delivery/Deployment (CI/CD) create an automated pipeline that takes code from the SSoT (Git), then compiles, tests, and deploys it—“untouched by human hands.” This engine isn’t just for applications; it can automatically build infrastructure, test documentation for broken links, enforce security policies, and publish updates.



“A company saved ‘97-98% of testing time on multiple enterprise applications compared to manual testing” by integrating it into the CI/CD pipeline.”

- Citizant

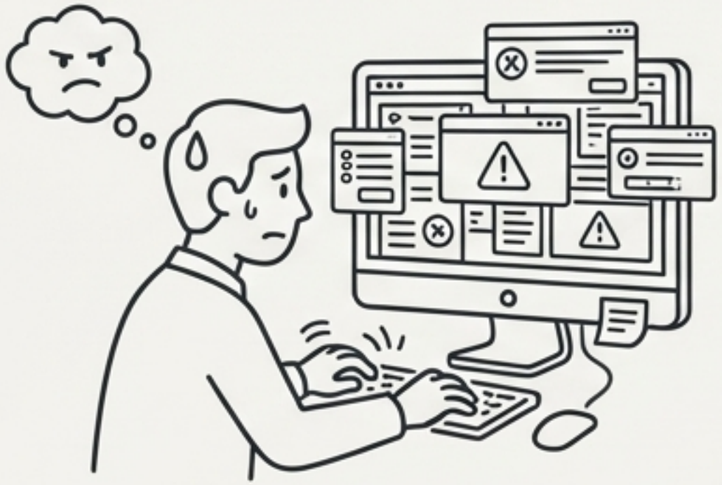
Application 1: Infrastructure as Code (IaC)

IaC is the practice of managing and provisioning infrastructure (servers, networks, databases) through machine-readable definition files, rather than manual configuration.

The ‘Multichannel” Mess

Pain Points: Manual server setup, long wait times, inconsistent environments (“it works on my machine!”), and “snowflake” servers that are impossible to replicate. YAML can be “crazy” and “deceptively simple,” leading to errors like the infamous “Norway problem.”

Real-World Cost: Manual server provisioning took **34 days**.



Snowflake Server

The “Omnichannel” Blueprint

Solution: Tools like Terraform or Ansible use code to define infrastructure. A single script can deploy identical, repeatable environments in minutes.

Real-World Win: Server provisioning time reduced from **34 days** to **7 minutes**.

```
main.tf

resource "aws_instance" "web" {
  count         = 3
  ami           = "ami-12345678"
  instance_type = "t2.micro"
}
```

Consistent, repeatable environments

Application 2: Documentation as Code (Docs as Code)

Docs as Code treats documentation like software. It lives in the same repository as the product code, is written in a simple markup language like Markdown, and is reviewed and versioned through the same pull request workflow.

The ‘Multichannel’ Mess

Pain Points:

Documentation lives in scattered Word docs, wikis, or shared drives. It’s quickly outdated, hard to find, and disconnected from the actual product code.

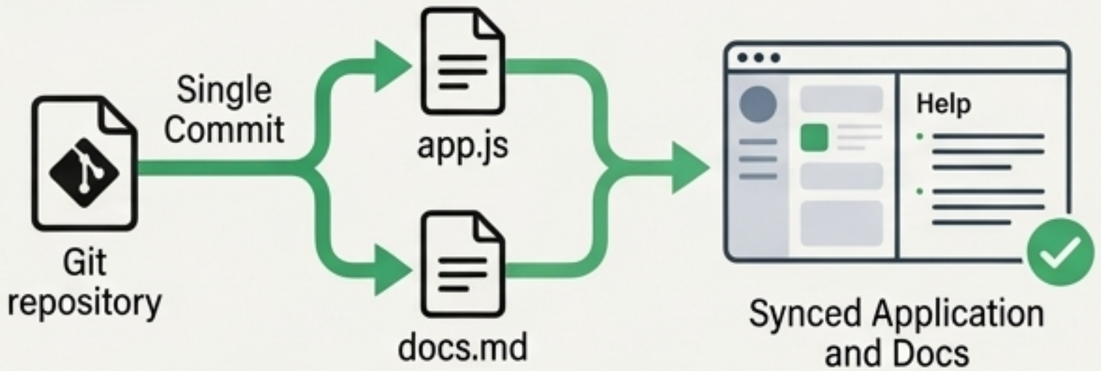
“Imagine you’re reading a user manual... On page one, the button you need to click is called **OK**. On page three, it’s called **Confirm**. And on page five, it’s just ✓. Confusing, right?”

– Jayana Saldanha, Sardine

The ‘Omnichannel’ Blueprint

Solution:

Documentation and code are updated together in the same commit. The CI/CD pipeline automatically tests for broken links and deploys the docs to a static site (using generators like Hugo, Jekyll, or Docusaurus). This creates a Single Source of Truth for both the product and its instructions.



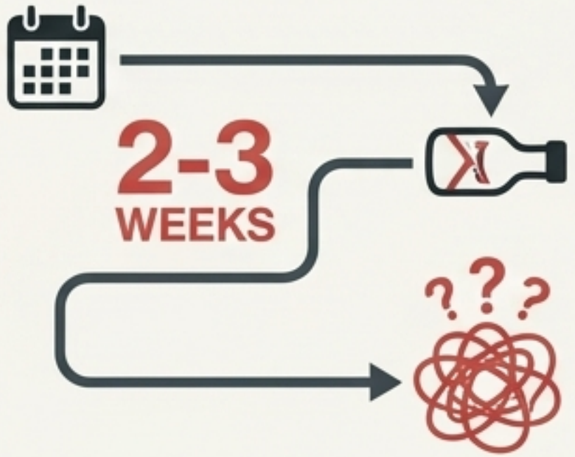
Application 3: Governance as Code (GaC)

GaC is the practice of defining policies for security, compliance, and operations as code. Instead of manual reviews and long checklists, rules are written in a declarative language and enforced automatically.

The ‘Multichannel’ Mess

Pain Points: Governance relies on manual processes, documentation, and slow review cycles. It’s often a bottleneck at the end of development.

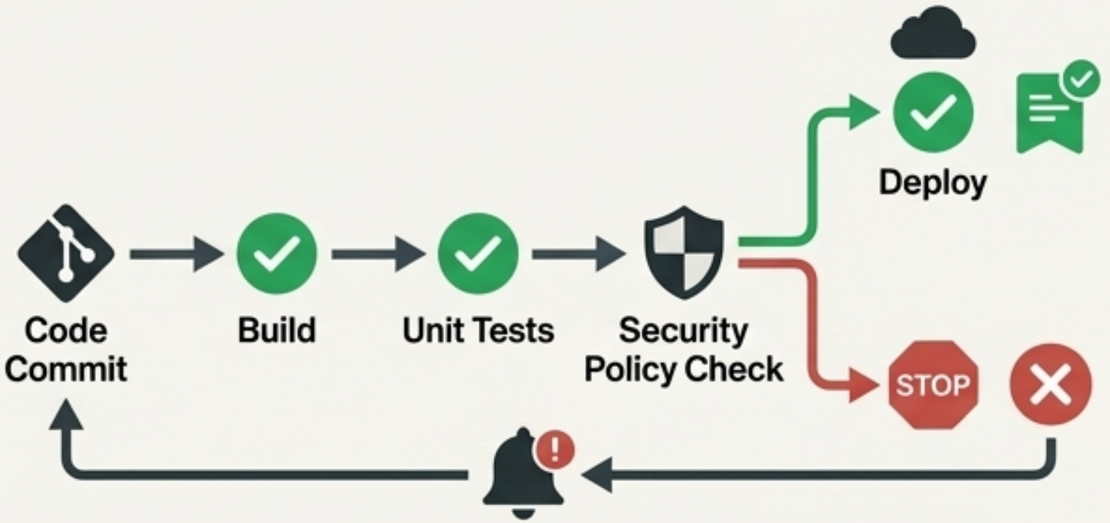
Real-World Cost: A manual security scan request could take **2-3 weeks** to generate a report, causing expensive delays.



The ‘Omnichannel’ Blueprint

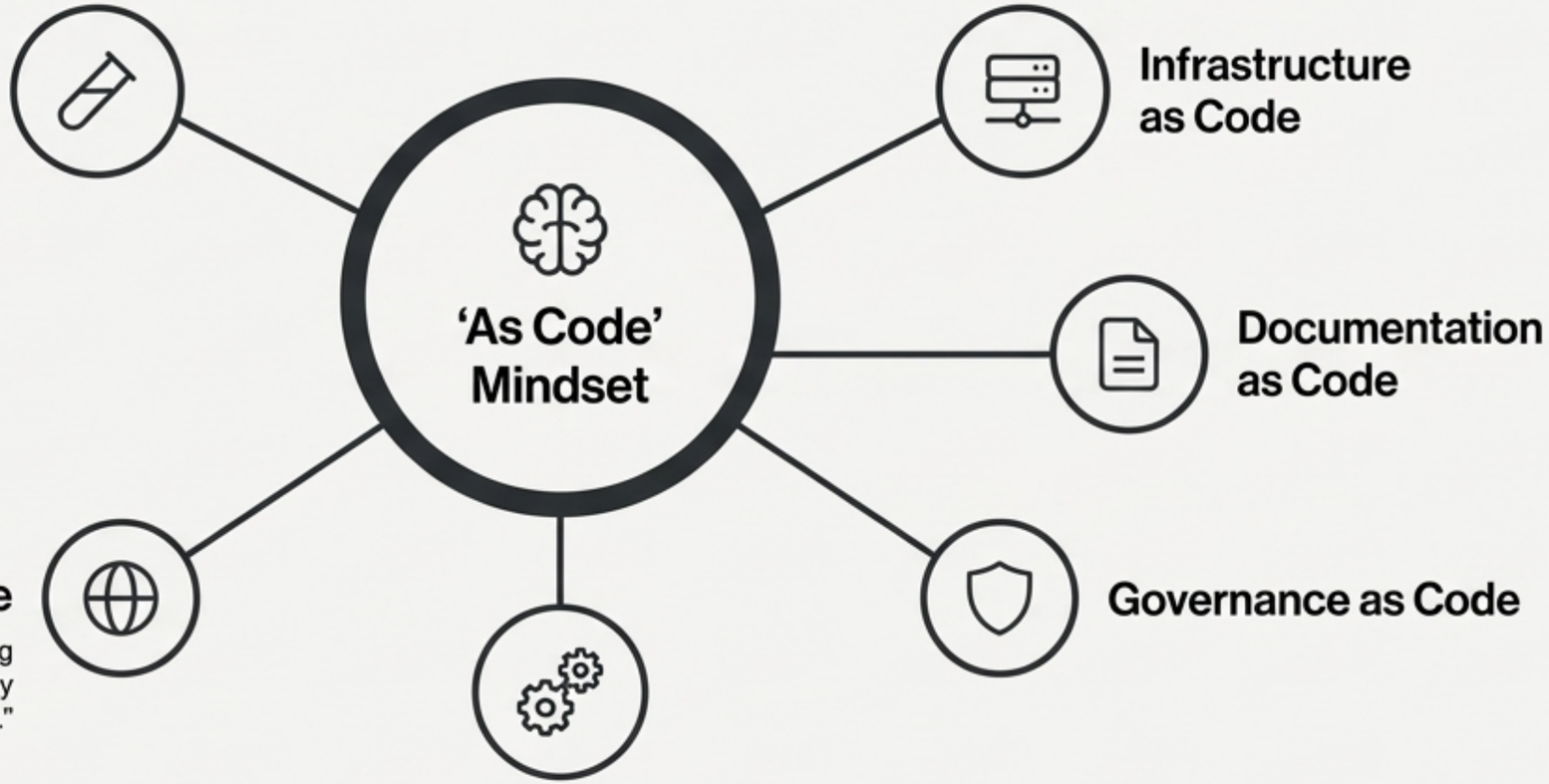
Solution: Policies are written once and automatically applied across all environments via the CI/CD pipeline. Each stage has gated compliance thresholds; if a check fails, the pipeline stops.

“Compliance as Code is baked into Citizant’s CI/CD pipeline.”



The ‘As Code’ philosophy extends to every part of the lifecycle.

This model isn’t limited to a few disciplines. The core principles of version control, automation, and a single source of truth can transform any process into a reliable, repeatable, and scalable workflow.



How does this relate to ‘No-Code’? They solve different problems.

“Everything as Code’ and ‘No-Code’ are two powerful, but different, approaches. ‘No-Code’ democratizes development by empowering non-technical ‘citizen developers’ to build applications visually. ‘Everything as Code’ gives technical teams a systematic way to manage the entire lifecycle of complex systems. One is about ***building the app***; the other is about ***managing the system that delivers the app***.

No-Code Platforms

Primary User: Business Users, Marketers (‘Citizen Developers’)

Method: Visual, drag-and-drop interfaces.

Goal: Rapidly build functional apps and workflows.

Example Tools: Knack, Lobster.

‘Everything as Code’

Primary User: Developers, DevOps, SREs

Method: Text-based, code editors, version control (Git).

Goal: Automate and manage the entire delivery lifecycle.

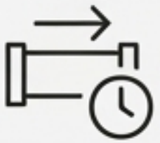
Example Tools: Terraform, Ansible, GitHub Actions.

The Omnichannel Payoff: Proven Results from a Unified Approach.

Xerox: Preventing Redundant Work

Saved over **$100,000,000**

Their “Eureka” knowledge-sharing system allowed an engineer in Brazil to solve a problem using a 90-cent part identified by an engineer in Canada, avoiding a **$40,000** machine replacement. A shared source of truth prevents reinventing the wheel.



IRS (via Citizant): Driving Massive Efficiency

20,000+ hours saved per year

Fully automated CI/CD pipelines for over 100 applications drastically cut down on manual effort. Containerization also led to potential savings of over **$4 million** by consolidating 3,000 servers to 180.

Geisinger Medical Group: Improving Critical Outcomes

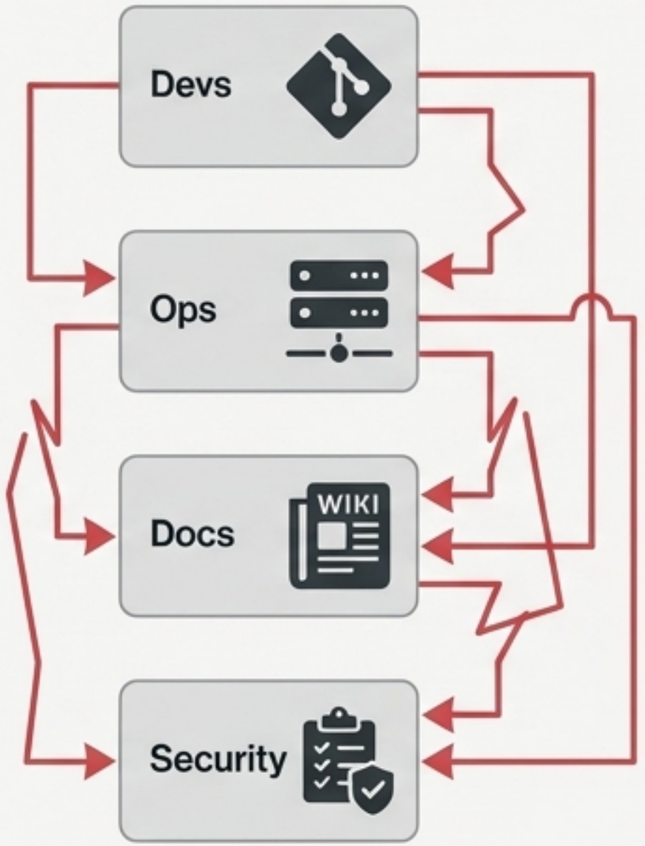
$2,000 savings per patient

Implementing a standardized 40-step surgical checklist—a form of “Procedure as Code”—removed variability, reduced complications, and lowered costs.

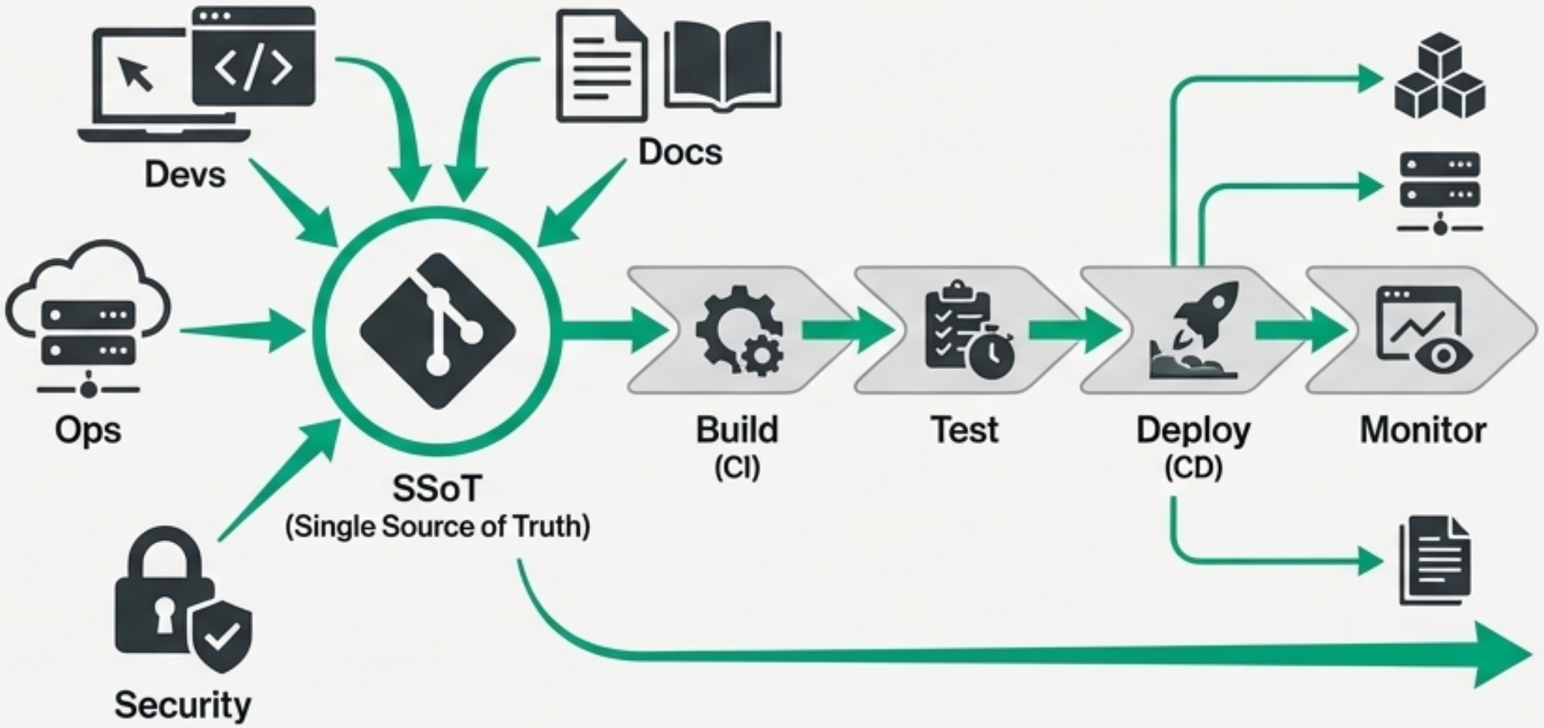
Neue Haas Grotesk Display Pro (65 Medium)

The journey from multichannel chaos to an omnichannel blueprint

Before: Multichannel Chaos



After: Omnichannel Blueprint



The ‘As Code’ Advantage: Velocity, Consistency, and Collaboration.

Adopting an ‘Everything as Code’ approach transforms your development lifecycle by embedding three key advantages into your workflow:

Velocity

Automation accelerates everything from testing to deployment.

Consistency

A Single Source of Truth ensures every environment and document is repeatable and reliable.

Collaboration

Shared, code-based workflows break down silos and unite teams around a common process.

“Knowledge management is nothing more than managing information flow, getting the right information to the people who need it so that they can act on it quickly.” – Bill Gates, *Business at the Speed of Thought*