Agentic Software Development

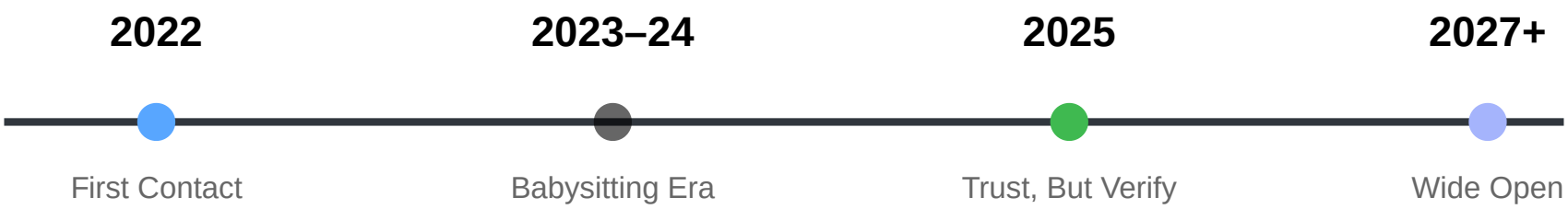
The Ubunatic Playbook

How one developer builds — and keeps alive — a fleet of focused, high-quality apps with AI agents.

Uwe Jugel · Ubunatic Coding

The Journey

From Babysitting to Verified Autonomy



From code-davinci in a private beta to local-first agent fleets.

- The turning point: stop **babysitting** every line, start **verifying** every result.

Core Principles

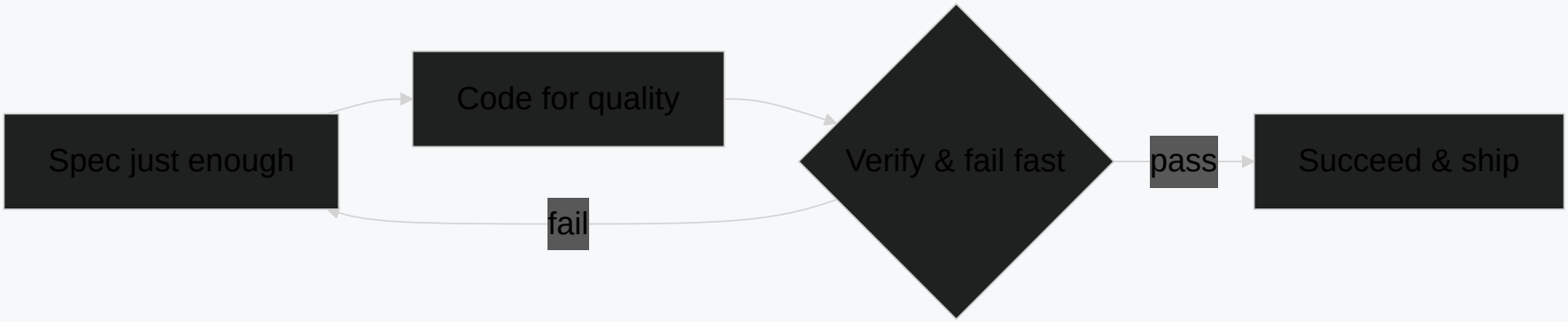
Spec Just Enough. Verify First.

Engineering Principle

Spec just enough, code for quality, verify and fail fast, iterate, succeed.

Agentic Principle

Don't automate the creation before you automate the verification.



Canary-First Development

Verify Environment Capabilities

- A **canary** is a minimal, standalone script that runs against the *real* environment.
- Detects OS quirks, toolchain limits, and sandboxing restrictions **before** feature code exists.
- Example: catching Snap confinement when printing PDFs with headless Chromium.

One Command

```
make canary
```

Verifies `mdbook`, `chromium`, and a full build → print → PDF round trip.

The Verification Gate

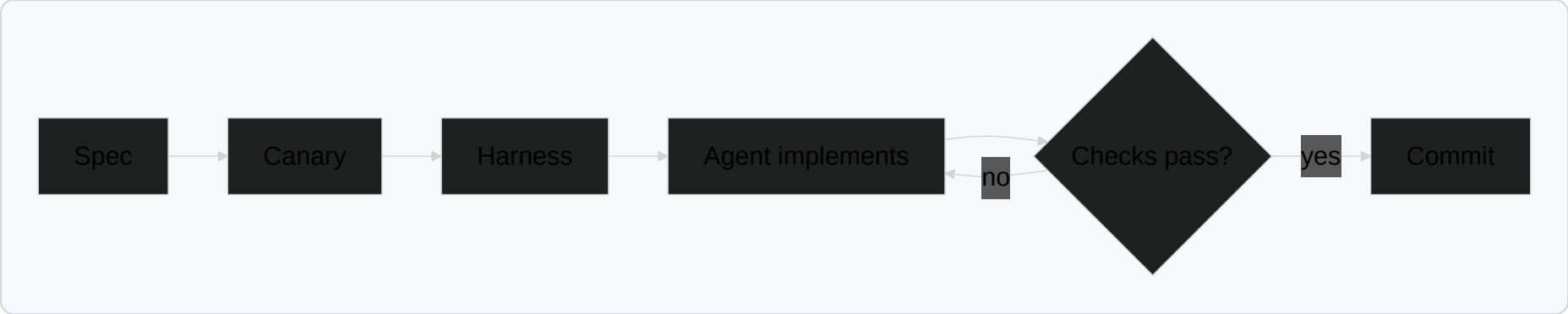
Numbers That Keep Us Honest

0	1	100%
unverified commits allowed	command to prove the toolchain	offline-first, no CDNs

> *Preflight mandate: always run `make preflight` before completing a task.*

The Golden Path

One Feature, End to End



- A real feature — book fonts and colours sourced from `spec/config.yaml` — followed all the way.
- The agent's first implementation **was wrong**: a quote-stripping expression mangled the font stack.
- The fix is only half the step. The other half is the test that would have caught it.

The Failure the Harness Missed

Green, and Still Wrong

- The goldens covered slide rendering. **Nothing covered spec parsing.**
- Every check passed. A human noticed the page looked wrong.
- The honest lesson is not "the harness works" — it is where the harness *ends*.

The Rule

Green and still wrong is exactly where the human belongs.

Add the check, then fix the bug — in that order, or the bug comes back.

Who Holds the Wheel

Three Plates, One Contract

🤖 Agent

Mechanical and verifiable. Delegate it.

👨‍✈️ Human

Taste, architecture, the irreversible.

🛡️ The Check

The exact command that catches the agent being wrong.

> *Every 🤖 in the book carries a matching 🛡️. A delegation without a check is not delegation — it is review debt arriving at machine speed.*

House Rules

A Subset Chosen for Readability

Go

Stdlib first. No ORMs. No global mutable state. Table-driven tests.

Zig

Zero-allocation hot paths. Explicit memory. Compile-time specs.

Bash

Header `set -euo pipefail`. Use `test`. Continuation alignment.

Make

Sentinel ⚙️ for phony rules. Self-documenting help targets.

- In a fleet read by a rotating cast of models with no memory, **reading is the dominant cost**.
- The subset is drawn on one test: can a reviewer — or a check — tell right from wrong quickly?
- Reference material, shelved as an appendix. Meant to be **grepped, not read**.

TUI Restoration & Safe Teardown

Surgical Precision on Raw Terminals

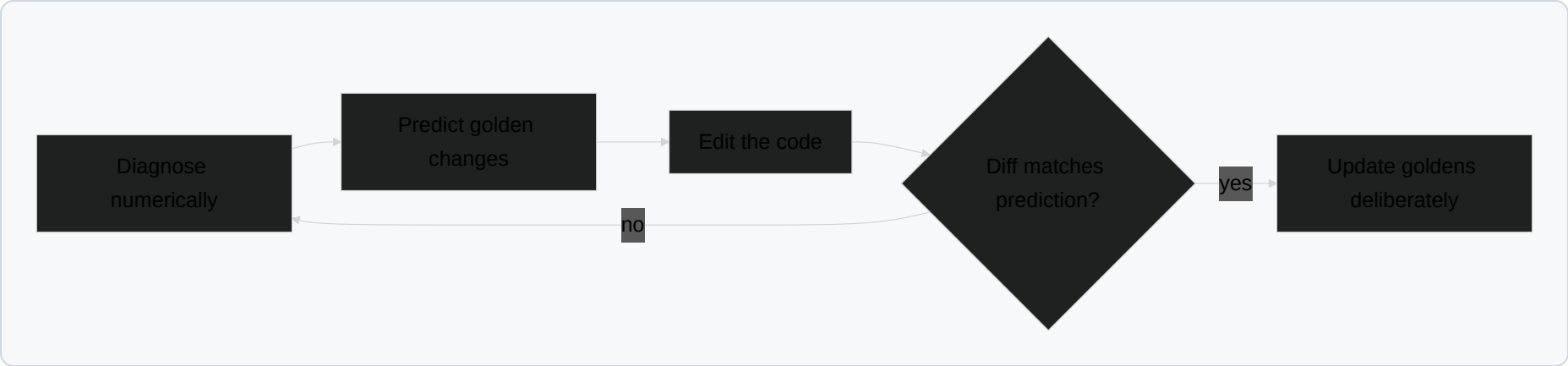
- **Exit Contract**: disable mouse, drain inputs, erase rows, park cursor.
- **Alt-Screen Trap**: avoid `?1049l` unless you entered the alt-screen.
- **DEC Private Modes**: the `?` is load-bearing (`?7l` vs `7l`).

The Exit Contract

```
\x1b[?1002l   mouse off
\x1b[6n       CPR drain
\x1b[2K       erase row
\x1b[?25h     cursor home
```

The Golden-Change Playbook

Proving Visual Regressions Safely



- **Anti-patterns**: never `-update` to make tests pass; eliminate downstream hacks.

Agentic Dev Patterns

Workflows Built for Model Strengths

Context Recovery

Write issues, errors, and plans to `issues/` before compaction.

Spec-Driven

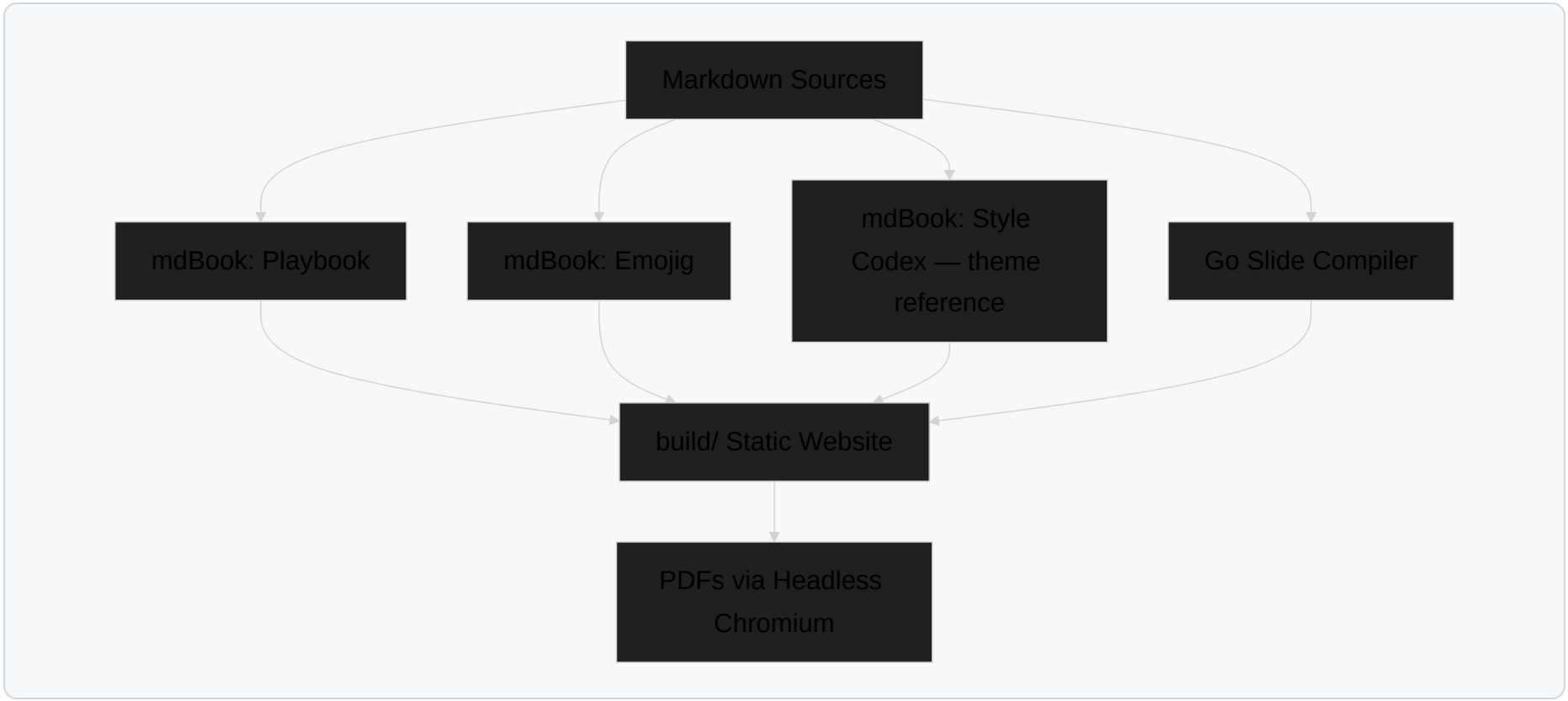
Define constants in specs, auto-generate derivatives.

Minimize

Avoid duplicate implementations across languages.

One Static Website

Books, Decks, and PDFs from One Build



- One `make build` renders everything. `make preflight` proves it: unit tests, theme goldens, and a link check that fails on a dead or non-portable link.

The Fleet Effect

Why This Compounds

n	1	0
apps maintained by one developer	verify command per repo	context lost between sessions

- Every repo greets its next visitor — human or agent — with the same files, checks, and rules.
- Correctness is checked by the **repo**, not remembered by the developer.
- Uniformity, not heroics, keeps the fleet alive.

Where It Breaks — and What to Steal

The Honest Chapter

Limits at Scale

Teams need trackers, governance, and meetings. Files alone don't page anyone at 3 a.m. — the solo purism does not transfer.

The Substrate Does

Verification-first gates, in-repo ADRs, agent-ready repos, one-command preflight, platform canaries, spec-driven config.

Full review: The Solo Fleet, Reviewed — in the book, and as its own deck.

Where to Start

Pick Your Budget

5 minutes

The Rule Index — every rule we hold ourselves to, one line each, with what enforces it.

15 minutes

The Golden Path — watch an agent build, fail a check, and recover.

Want proof?

Emojig Chronology — six weeks in Zig, where these rules were paid for in real bugs.

> *Everything is offline-first and local:* `make build` *renders the shelf,* `make preflight` *proves it.*