Manifesto for Efficient Verification Flows

From "Script Wilderness" to Professional Tool Architecture

Malte Henzelmann

February 28, 2026

Core Principles

1. Location Agnosticism

A script must never assume the directory from which it is called.

→ **The Rule:** A tool resolves all its dependencies relative to its own installation path, ensuring it works regardless of the user's current working directory.

2. Self-Containment (Ending "Sourcing Hell")

Manually sourcing environment variables pollutes the shell's global state and leads to conflicts.

→ **The Rule:** Wrapper scripts manage their environment internally. Upon completion, the user's shell remains clean and unaffected.

3. The "Two-Button Doctrine" (Standardized Interfaces)

Complexity belongs "under the hood," not in the user's manual.

→ **The Rule:** Every environment provides a consistent, unified interface: `build` and `run`. This ensures 1-minute-to-simulation for new team members.

4. Separation of Source and Artifacts (Clean Output)

Writing results into the source tree prevents clean version control and creates noise.

→ **The Rule:** All generated data is stored in dedicated output directories. The source tree remains "untainted" and ready for Git/SVN operations.

5. Robust Tooling over Shell Hacks

Shell scripts are for simple command chains, not for complex logic or path manipulation.

→ **The Rule:** As soon as logic (conditionals, loops) is involved, we use maintainable, testable languages like **Python** or **Dlang**.

Professional Honor

I never leave a tool environment in an unstable state. A flow is only "finished" once it works intuitively, stably, and reproducibly.