

The illusion of flawless code

What the compiler sees:

81 commits from 5 agents working perfectly. Code runs. Zero errors.



38 commits ahead.
0 TypeScript errors.

What actually shipped:

Dead `{custom_story}` template placeholders leaking into LLM prompts.



The system works, but it wastes tokens on every message.
Building is only half the job.

Generalists check for correctness, specialists hunt for vectors

Spawning four specialized reviewers in parallel surfaces fundamentally different bugs from the exact same files.

Pipeline Reviewer

- Audits index.ts & system prompts
- Tracing data flow across module boundaries

Security Auditor

- Probes for attack vectors
- API keys, SSRF, path traversal, prompt injection

Media Reviewer

- Validates multimodal pipelines
- Transcriptions, vision calls, file downloads

Onboarding Reviewer

- Inspects state machines
- Commands, schemas, preset configurations



A five-minute audit reveals a completely unshippable system

わア

4 Reviewers.
5 Minutes.
2 Blocks.
2 Warnings.

Triage Board

2 BLOCKs (Critical)

- **Path Traversal:** presetId directly in file path
- **Cost Attribution:** processMedia() runs before userId
- **Bot Token Leak:** Telegram URLs in error logs
- **Typing Indicator:** clearInterval missing on empty batches

10+ HIGH Issues

- Unbounded reference caches
- Missing timeouts on generation
- Template injections via \$
- Missing MIME validation
- Complete rel cache a-m validation

Parallel fixes demand strict, non-overlapping file ownership

Meanwhile, a fourth parallel agent updates CLAUDE.md and SCHEMA.md. No two agents ever edit the same file.



Security Fixer

Owens: `agents.ts`, `file-download.ts`

Task: Add allowlist validation;
scrub token leaks.

Pipeline Fixer

Owens: `index.ts`, `loop.ts`

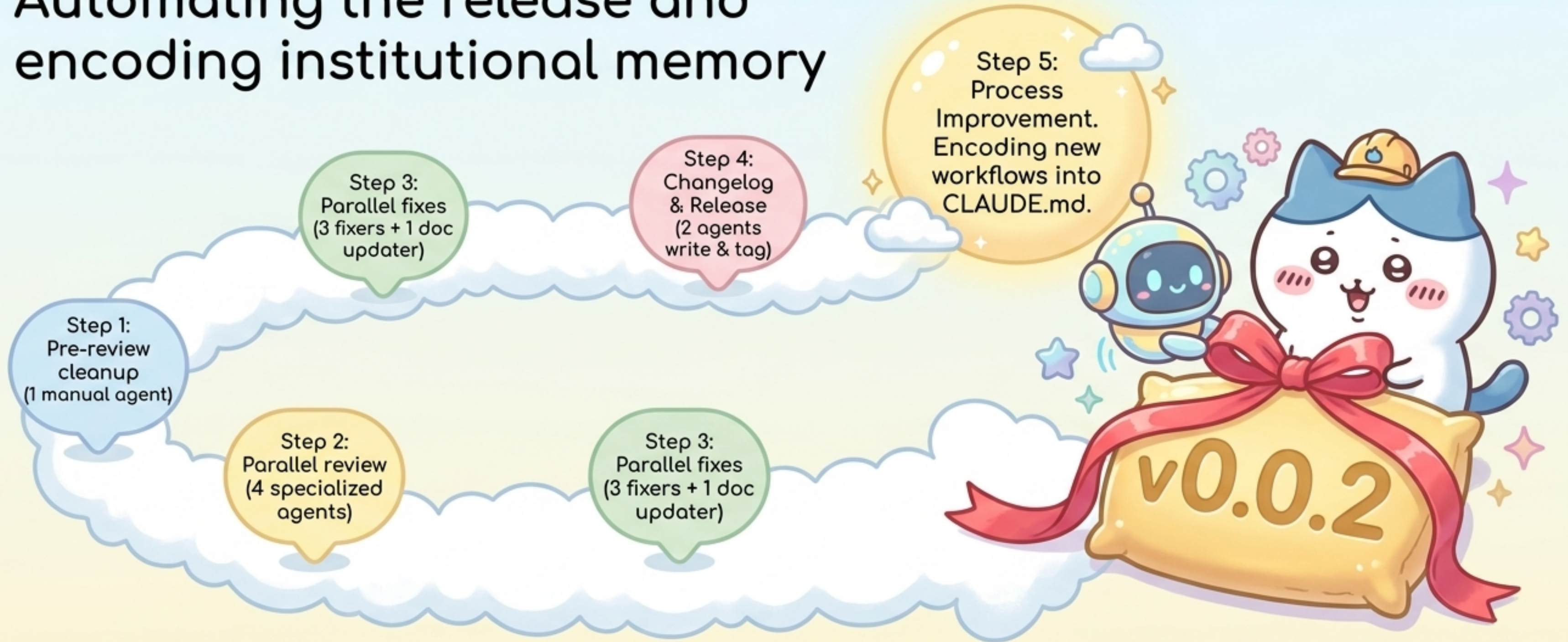
Task: Fix typing indicator;
patch cost attribution.

Onboarding Fixer

Owens: `commands.ts`, `onboarding.ts`

Task: Resolve Promise drops;
fix race conditions.

Automating the release and encoding institutional memory



Building with AI is the start. Shipping with AI requires institutional memory that persists across context windows.