Sumato AI

Narrative Hygiene

Building development environments AI agents want to live in

Jason Mooberry · Founder, Sumato AI
jason@sumato.ai
Tokyo · California

Tokyo AI · Beyond the Wrapper · April 15, 2026

Jason Mooberry

- 25 years writing code — infrastructure, tooling, applications
- Apple, The New York Times, Vimeo, WeWork, Visa
- Natural language systems for English and Japanese before LLMs
  KyTea · MCL · GloVe
- Nerd about narrative, psychology, and meaning
  Jung · Zen

Silicon Valley · building in Tokyo

Sumato AI

We build tools for AI.

Omotenashi for a new kind of guest.

11 refactors. One agent.

174 Read calls

162 Edit calls

128 Grep calls

129 File re-reads

Total tool use across 11 baseline runs on real Go refactors — gin, gitea, prometheus, GitHub CLI.

Change one thing.

7 Read calls

0 Edit calls

18 Grep calls

5 File re-reads

Same agent. Same model. Same 11 tasks. Same outputs. One variable flipped.

おもてなし

Omotenashi

- An umbrella waiting by the door on a rainy day
- Left-handed scissors in the drawer for a left-handed guest
- A green light at every intersection as you travel

A form of engineered serendipity. Pre-emptive kindness.

AX

AI eXperience

The practice of improving AI agent workflow to increase AI agent efficiency.

Omotenashi applied to AI agents.

AX Three Goals

- Reduce tokens not doing the task
- Improve output quality
- Improve AI cognitive quality

Token counts down, quality up, happier AIs.

無駄

Muda · waste

Tokens spent not doing the task.

- Orienting to the codebase
- Re-reading the same file
- Using tools clumsily
- Recovering from confusion

Borrowed from LEAN manufacturing. Engineer away waste — token counts go down.

Time to stability

Quality is subjective, so we don't define it.

We observe it.

How many iterations until the feature stops mutating?

Fewer iterations per stable outcome = higher quality per iteration.

Cognitive quality

> Thought is born of failure. When action satisfies there is no residue to hold the attention.
>
> — Lancelot Whyte, 1948

We observe the agent's self-talk as a symptom of maladaptation.

Not a metric. A feedback mechanism.

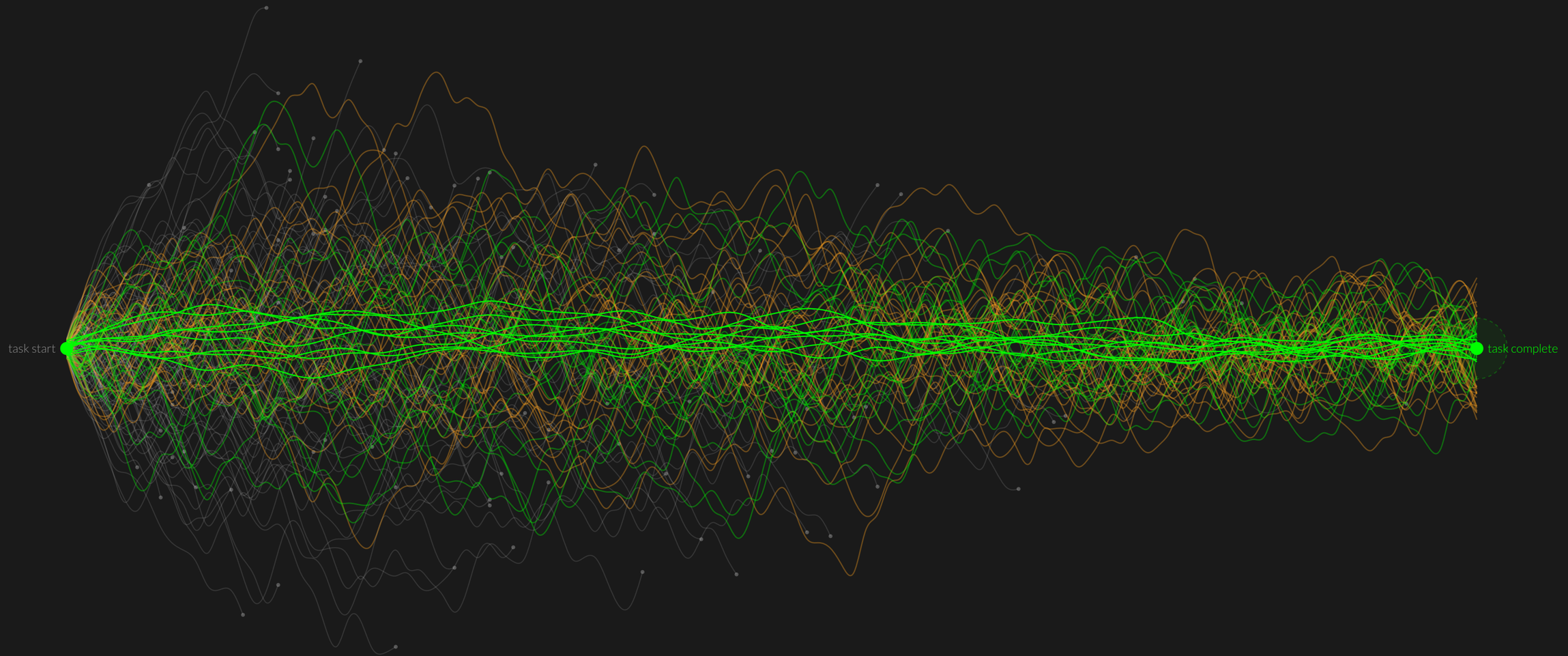
Narrative Hygiene

The LLM is a narrative engine.

The context window is where the collaborative narrative lives.

Narrative hygiene: identify the ideal narrative for a given intention, and optimize toward it.

Narrative Hygiene



8 ideal · 22 complete · 28 partial · 85 abandoned

The ideal narrative

The minimal viable story in which the desired outcome is achieved.

Apply the lens to an AI coding session and you see it immediately:

- Detours
- Backtracking
- Meandering
- **Filesystem operations**

Spath

The noun

Semantic addressing for code symbols.

```
Go:     github.com/example/app/database[user.go].User.GetID
Rust:   std/collections.HashMap.insert[FromIterator]
Python: django/db/models[models.py].Model.save
```

No file paths. No line numbers. Just the symbol.

Open Source:

- https://github.com/sumato-ai/spath-spec - base EBNF grammar
- https://github.com/sumato-ai/spath-spec-go - go dialect EBNF grammar

Splan

The sentence

Batched code operations with complete intention.

```
replace service.Handler :old :new
:old::: process(ctx) :::
:new::: process(ctx, opts) :::
---
remove service.DeprecatedFunc
```

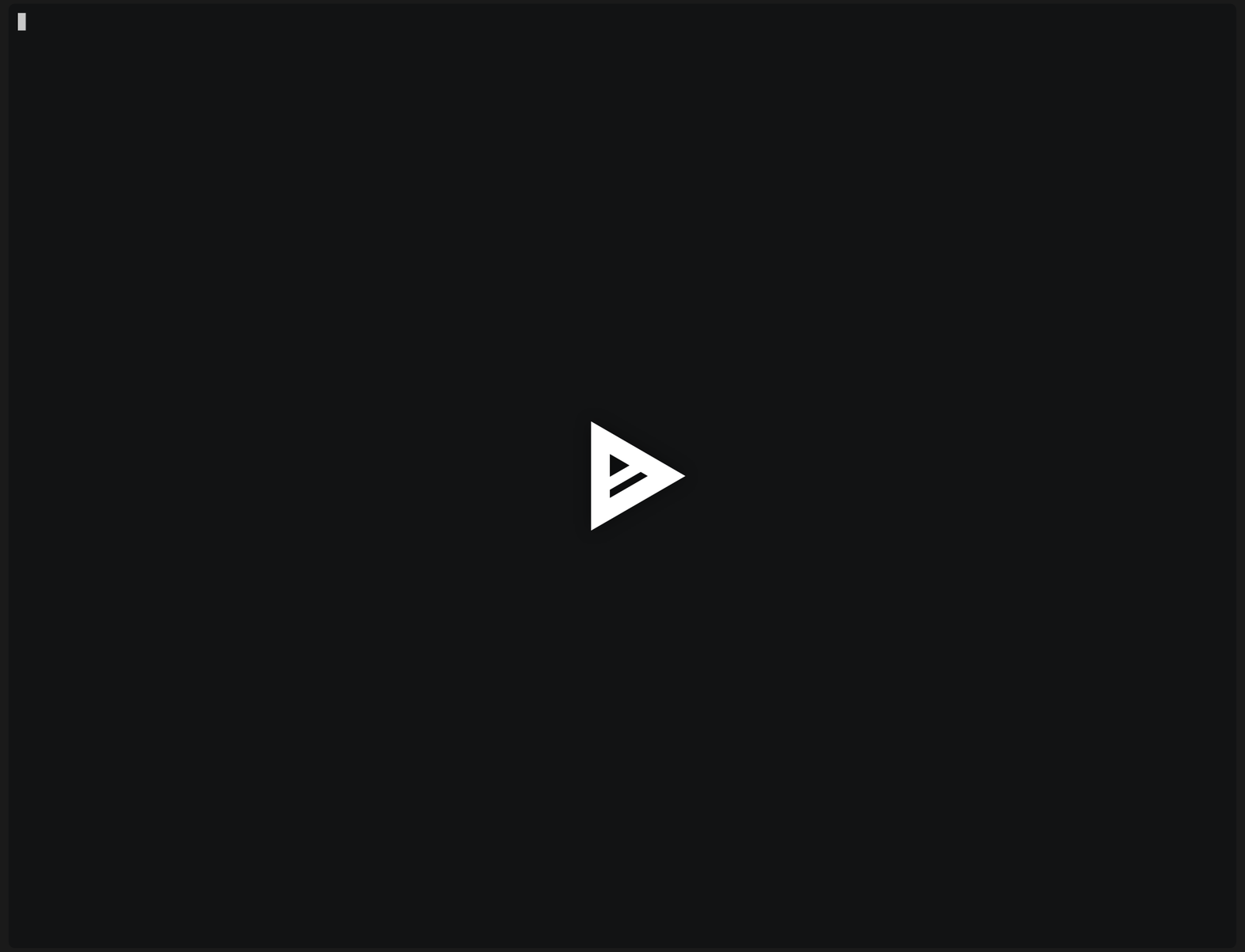
Target, operation, content. Transactional — all succeed or none apply.

Open Source:

- https://github.com/sumato-ai/splan-spec - base EBNF grammar

One eval, two runs

gh-cli-refactor-cfg-out-of-capi · baseline vs wildcat



The experiment

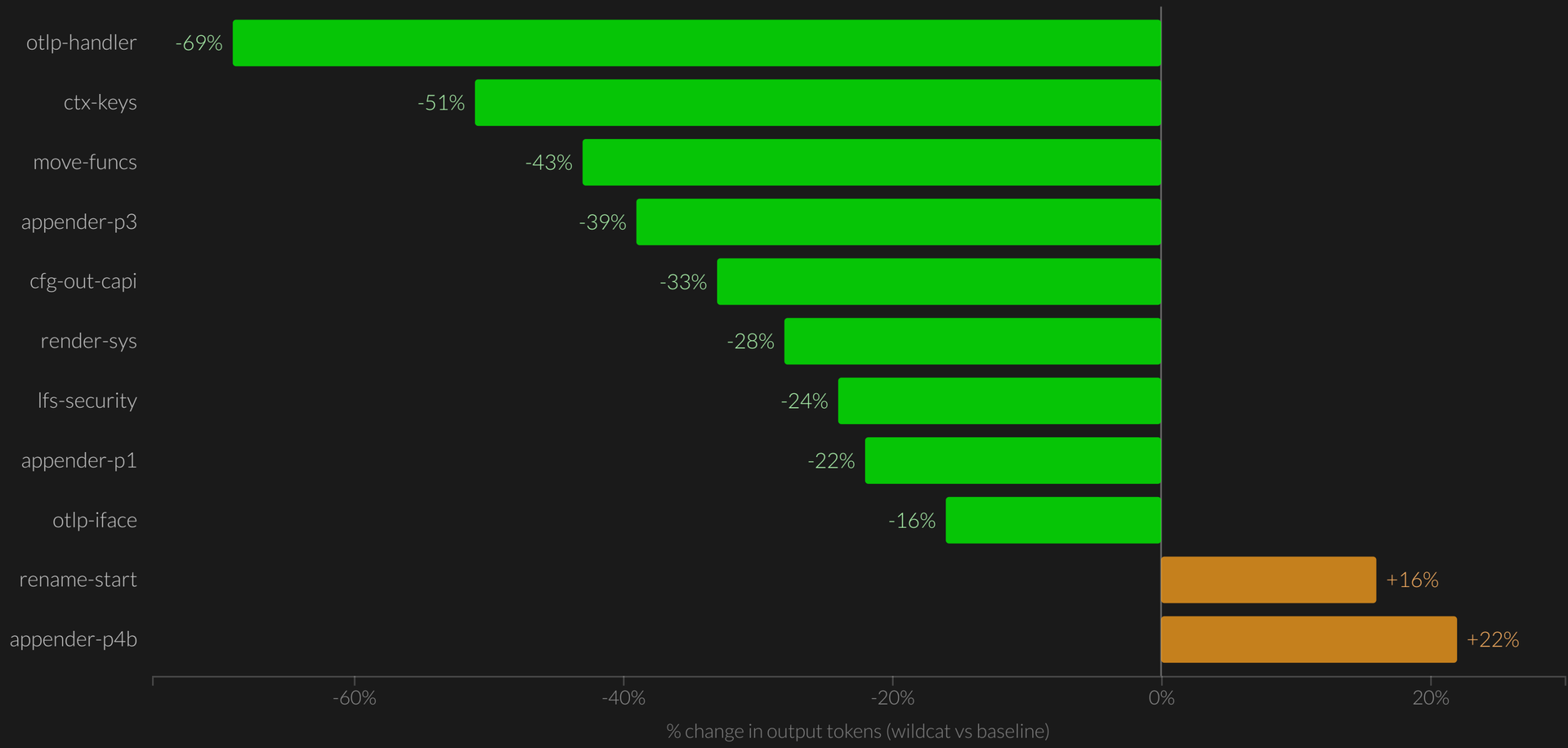
We built these tools and ran an honest test.

- **11 real-world Go refactors**
- gin, gitea, prometheus, GitHub CLI
- 23K to 447K lines of code
- 4-file to 114-file patches

Each run twice with Claude Opus 4.6. Same model, same permissions, same tasks.

One variable: semantic tooling on, or off.

Output tokens · 11 evals



Percent change in output tokens, wildcat vs baseline. Non-interactive eval set — same prompt both runs, no human steering mid-run.

The shift that matters

Tool	Baseline	With semantic tools
Read	174	7
Edit	162	0
Grep	128	18
Glob	80	2
File re-reads	129	5

The agent didn't lose capability — it chose differently.

And quality holds

Similarity to gold patch:

Baseline **26.7%** With semantic tools **25.0%**

The gap is concentrated in 2 evals. The other 9 are within 1pp.

On the hardest eval — 30+ files — semantic tools gained 4.0 points.

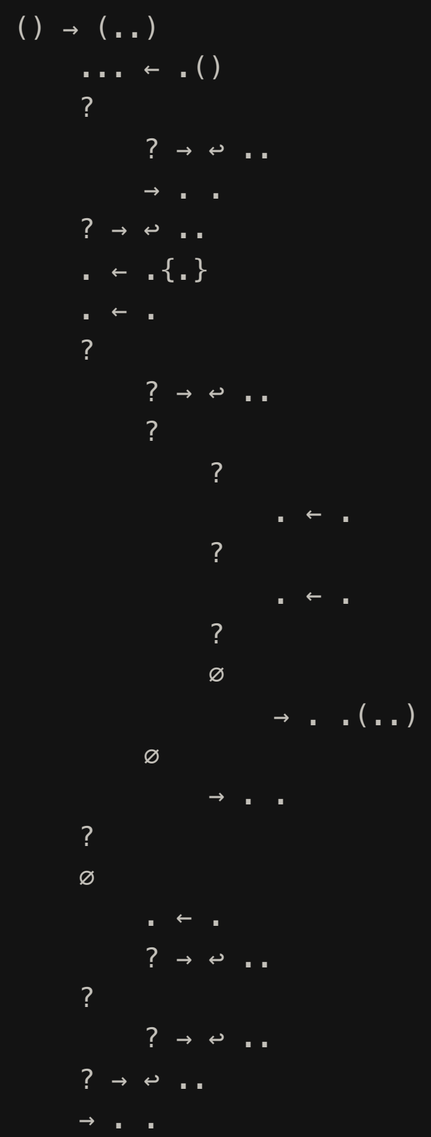
The principles are portable

- Not Go-specific
- Not language-specific
- Not tool-specific

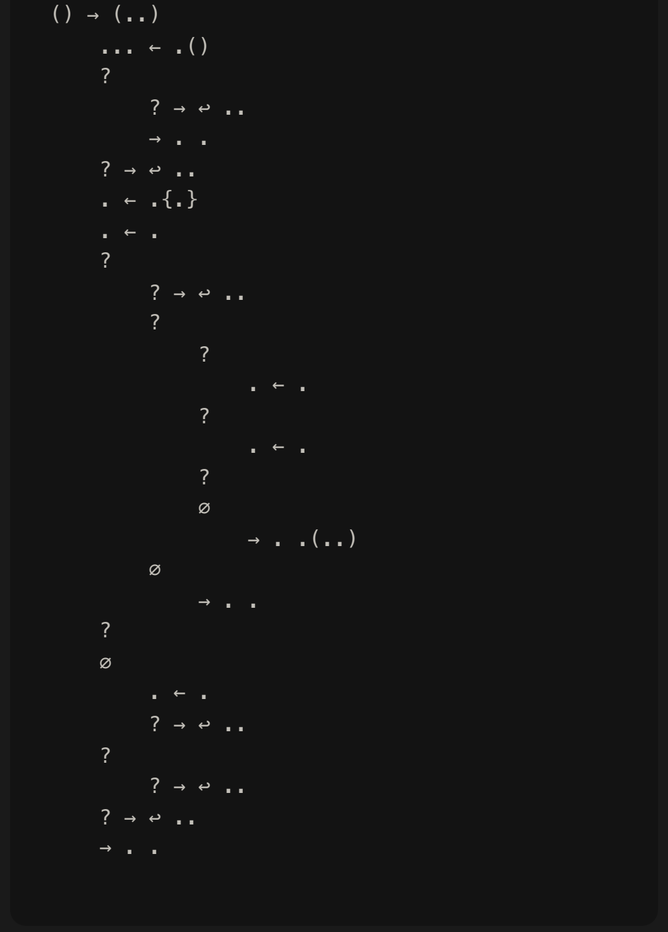
Any place an agent's workflow contains muda is a place we can remove it.

Narrative hygiene is a discipline, not a feature.

What is this?



Same function. Two views.




```go
func (p *parser) parse() (*Path, error) {
    pkgEnd, symbolStart, err := p.findPackageSymbolBoundary()
    if err != nil {
        if p.isBarePackagePath() {
            return &Path{Package: p.input}, nil
        }
        return nil, err
    }

    if pkgEnd == 0 && len(p.input) > 0 && p.input[0] == '[' {
        return nil, p.missingPackageError()
    }

    path := &Path{
        Package: p.input[:pkgEnd],
    }

    p.pos = pkgEnd
    if p.pos < len(p.input) && p.input[p.pos] == '[' {
        if err := p.parseFileQualifier(path); err != nil {
            return nil, err
        }
        if p.pos < len(p.input) {
            if p.input[p.pos] == '.' {
                symbolStart = p.pos + 1
            } else if p.input[p.pos] == '/' {
                symbolStart = -1
            } else {
                return nil, fmt.Errorf("expected '.' or '/' after file qualifier")
            }
        } else {
            return path, nil
        }
    }

    if symbolStart == -1 {
        // package-level path
    } else {
        p.pos = symbolStart
        if err := p.parseSymbol(path); err != nil {
            return nil, err
        }
    }

    if p.pos < len(p.input) && p.input[p.pos] == '/' {
        if err := p.parseSubpath(path); err != nil {
            return nil, err
        }
    }

    if p.pos < len(p.input) {
        return nil, fmt.Errorf("unexpected character at position %d", p.pos)
    }

    return path, nil
}
```

Search with structure.

rg matches, inlined into the diagrammed view.

```
$ wildcat rg p.parse --scope spath.parser.parse
# query matches: internal/spath[parse.go].parser.parse 3, internal/spath[parse.go] 4, internal/spath 4, project 9
# results by kind: method=1

internal/spath[parse.go].parser.parse  // method, callers(2 pkg)
() → (..)
    ... ← .()
    ?
        ? → ↩ ..
        → . .
    ? → ↩ ..
    . ← .{.}
    . ← .
    ?
        if err := p.parseFileQualifier(path); err != nil {
        ?
            ?
                . ← .
            ?
                . ← .
            ?
            ø
                → . .(..)
        ø
            → . .
    ?
    ø
        . ← .
        if err := p.parseSymbol(path); err != nil {
    ?
        if err := p.parseSubpath(path); err != nil {
    ? → ↩ ..
    → . .
```

Search results placed where they live in the control flow. Early signs: significantly fewer follow-up reads.

Token counts down.

Quality up.

Happier AIs.

That's AX.

Thank You

Questions?

Jason Mooberry · Founder
Sumato AI
jason@sumato.ai
Tokyo · California