I tested 112 agent-dev resources. A plain setup won.

Skills, hooks, loops, routers and design packs. What I measured, what I threw out, and the rules worth copying.

112
catalogued

23
researched in depth

8
sandbox trials

Six things to take away

01 **Use one planner,** and let everything else plug into it as a part.

02 **"Done" means proven,** with the checks run and the evidence pasted.

03 **Price every add-on** by measuring its startup tokens before you install it.

04 **Prefer native features,** because the tool shipped them more often than not.

05 **Vet third-party skills like code,** by scanning them before they load.

06 **Expect most add-ons to lose,** since they failed their own trial or cost more for no gain.

How the test worked

112 → **23** → **8**

items catalogued

researched in depth

trials in a sandbox

- Each trial ran headless on Sonnet 5.5 in a throwaway HOME, against my real baseline setup.
- 44 runs in total. Token counts were cross-checked against the session transcripts.

The promotion rule

A candidate only gets installed if it clears all three.

01 **It beats the baseline** on the metric it claims to improve.

02 **It adds under about 2k tokens** to every session start.

03 **It has no open high-risk finding** from the security read-through.

> `Result: 0 candidates were promoted.`

METHOD

The baseline that won

Plain, boring, already installed.

superpowers

brainstorm, plan, build, review, verify, finish

AGENTS.md + shared skills

one rule file and one skills folder for every agent

rtk

compresses shell output before it reaches the model

Context7

current library docs instead of memory

ccusage status line

cost and context, always visible

```
Startup context in an empty folder:
```

Keep exactly one planner

0 of 23

researched resources were worth installing whole.

- Every kit that wrote its own plan, spec or task order was cut down to one part.
- Parts that survived: a review step, a gate hook, a read-only audit, a rule.

The pipeline

1	**Clarify**	ask until the requirements are settled
2	**Spec**	write it down and get a yes
3	**Plan**	short task list with exact files
4	**Build**	one task at a time, test first
5	**Verify**	run every check, paste the evidence
6	**Finish**	review, then merge or open the PR

Ask before you build

Beyond a small fix, nothing gets built until you approve a plan.

01 **Test first** for logic changes: red, then green.

02 **Build only what was asked,** with no options for later.

03 **One git worktree per feature** when several run at once.

Write down what "wrong" looks like

Every spec ends with one box per outcome. Each box is a yes-or-no test.

```
## Acceptance checks
Check files: tests/*.test.ts

- [ ] AC-1 · CHECK: npm test
       · EXPECT: exit code 0
       · EVIDENCE:
- [ ] AC-2 · CHECK: open /pricing
       · EXPECT: 3 plans visible
       · EVIDENCE:
```

After every decision, ask one question

"How would we know this came out wrong?" The answer becomes a check.

Testable

a command or test with a plain yes or no

Taste

you look at it and judge it against a stated rule

Undecided

a reviewer must answer CANNOT JUDGE until you decide

"Done" means proven

01 **Run every check yourself.** Tick the box and paste what you saw.

02 **Say how many checks passed,** for example "N of N checks pass".

03 **An outcome you will not meet** gets an ABANDON line and a note to the user. Never silence.

The reviewer never sees your summary

Give the final review the diff and the acceptance checks. Nothing else.

01 **A builder's summary primes the reviewer** to agree with the builder.

02 **The checks are the standard.** They were written before the code.

Cap the loop

40

subagents: the most one run may spawn unless you ask for more.

- At most 2 review-and-fix rounds.
- If the same finding shows up twice, stop and ask.

Lock the checks, then attack the lock

7 / 7

tamper attempts blocked after one fix.

- Approve the spec, commit it, tag it. A hook blocks "done" if a check changed after the tag.
- First run: 6 of 7 blocked. Re-tagging onto a weaker check got through.
- Still open: deleting the hook's own state file. That needs an OS-level write deny.

Tried, measured, set aside

Verification add-ons, against the plain baseline.

Plan review by a fresh agent

6/6 seeded defects found by both arms. 2.2x the cost.

REJECTED

Acceptance-gate default loop

6/6 outcomes in both arms. 1.35x the cost.

OPT-IN

3-vote skeptic filter on reviews

+7 points of precision. 11x the cost.

REJECTED

One rule file, every agent

01 **One AGENTS.md** is the source of truth. Each agent gets a link to it.

02 **One skills folder.** Install once, link everywhere.

03 **Skills over giant prompts.** A skill costs a description until it is used.

Native feature beats add-on

Second opinion mid-task

built-in advisor, not a hand-made agent

Parallel teammates

native agent teams, per session only

Huge command output

Claude Code saves it to a file; rtk trims the rest

Symbol search

serena, switched on per repo

Checking third-party skills

SkillSpector static scan

Code-navigation tools cost tokens

+29%

tokens for Graft on a ~100-file repo, versus plain search.

- serena used +7.7%. All three answered 5/5 tasks correctly.
- Startup cost: Graft +5,525 tokens, serena +1,881 tokens.
- Larger repos may differ. This trial did not test them.

Measure startup context

18.2k

tokens before the first prompt, in an empty folder.

- Last measured 16.5k. The extra ~1.7k came from new MCP server instructions, not from my changes.
- How: claude -p "/context" in a fresh directory.

Every add-on has a startup price

Tokens added to every session start.

Plan-review skill	**+35**
Slop-severity rubric	**+53**
Acceptance-gate hook	**+354**
Native advisor	**about +1,340**
serena MCP	**+1,881**
Graft	**+5,525**

> One eval kit as shipped: about 6.5k, or +40% on the baseline.

Trim output, not intent

01 **rtk rewrites shell commands** so `git status` becomes `rtk git status` with compressed output. Confirmed firing in a fresh session.

02 **Claude Code already saves** MCP results over 50k characters and Bash output over ~30k to a file. Do not add a second offload layer.

The silent kill switch

Any of these, even set to 0, turns off feature-flag fetching. The advisor goes with it.

```
DO_NOT_TRACK
DISABLE_TELEMETRY
CLAUDE_CODE_DISABLE_NONESSENTIAL_TRAFFIC
```

> A feature "not working" can be an environment variable.

System first, then screens

01 **Obey the project's DESIGN.md.** No file? Write one before any screen.

02 **Ground each new screen.** Name the app shell, the nav, the closest existing screen, the icon set and the logo file.

03 **Performance fixes** never remove the signature animation.

Slop is a cluster, not one CSS trick

Three templates. One tell alone is not slop.

Default SaaS

indigo gradients, glass cards, hype verbs

Editorial costume

big serif, tiny tracked mono labels, invented heritage

Tasteful middle

calm neutrals, one stripe, "all-in-one" copy

Grade the cluster, then the copy

Swap in another product's name. If every line still works, the copy is the problem.

High

three tells from one template, and generic copy

Medium

two tells, or a full template look with specific copy

Low

a single tell in an otherwise specific design

My own rubric lost

4/5

with the new slop rubric, against 5/5 for the plain critique.

- It downgraded editorial slop to Medium.
- A rule that sounds smart can make a working check worse. Run the trial.

What turned up in the wild

01 **Install scripts piped into a shell.** Found quoted in install docs.

02 **Setup scripts that rewrite global settings,** and can truncate settings.json.

03 **A nested agent run with bypassPermissions.** Nothing asks before it acts.

04 **Telemetry on by default.** One tool had 17.6K events queued to send.

Unattended loops cost real money

2,214

subagents: the worst case of one review loop at its defaults.

- Its prompt pre-approved the cost: "proceed without a cost warning".
- A first round alone could spawn 62 to 112 agents.

Vet before you install

The baseline had no check on third-party skills. That was the biggest gap.

01 **SkillSpector v2.13.3 static scan** returns SAFE, CAUTION or DO_NOT_INSTALL in seconds, with no model call.

02 **Only dependency names leave the machine.**

03 **My own trial of it was skipped.** Treat it as a plan, not a result.

Trial in a throwaway home

01 **Back up your config first.** Then try the add-on in a sandbox HOME.

02 **Diff the real config afterwards.** Mine: unchanged after all 8 trials.

03 **Extracted code is untrusted.** Never run it. Read it. Run Python with -I.

Pin the version, skip auto-upgrades

0.18.0

is the version I kept Graft on.

- From 0.22 it added a team cloud, sign-in and weekly upgrade nudges.
- Upgrade only to a version you have read.

Agent teams: per session only 54k

tokens spent before work starts, for a 3-teammate team.

- Each teammate pays the full startup context.
- Turn teams on globally and every named subagent becomes a teammate, which silently changes how your plans run.
- Use them for research, multi-lens review and competing-hypothesis debugging.

File ownership in every brief

01 **Files you own.** The implementer may edit these.

02 **Files you may read.** Everything else is read-only.

03 **Implement one task at a time.** Parallel implementers collide on the same files.

Cheap models do the legwork

List prices per million tokens (input / output), cached on 2026-10-06 and may have changed.

Haiku 5.5

$0.10 / $0.50 search, file discovery, extraction

Sonnet 5.5

$2 / $10 routine builds, tests, focused reviews

Opus 5.5

$4 / $20 planning, judging, synthesis

Scoreboard 1 of 2

Fresh-agent plan review

6/6 defects, both arms. 2.2x cost. The "blank" reviewers saw my rules.

REJECTED

Answer-key question

3/3 defects, both arms. No gap to close.

TIE

Acceptance-gate loop

6/6 outcomes, both arms. 1.35x cost.

OPT-IN

3-vote skeptic filter

Precision +7 points. 11x cost.

REJECTED

Scoreboard 2 of 2

Graft for navigation

5/5 correct. +29% tokens.

REJECTED

serena by default

5/5 correct. +7.7% tokens. Keep it per repo.

OPT-IN

Native advisor

3/3 root causes, both arms. It was called 0 times.

TIE

Slop-severity rubric

4/5 against 5/5 for the plain critique.

REJECTED

Read this before you copy me

01 **The tests were too easy.** The baseline hit full marks on 5 of the trials, so no add-on could show a gain. Harder tasks might.

02 **One person, one model.** Sonnet 5.5, one setup, one machine.

03 **Prices are dated.** Cached on 2026-10-06, so check before you budget.

COPY THIS

10 rules for your AGENTS.md (1 of 2)

```
1) One planner: clarify, short plan,
   approval, then build.
2) Test first for logic changes.
   Prove it before saying done.
3) Build only what was asked.
4) Every spec ends with acceptance
   checks: CHECK, EXPECT, EVIDENCE.
5) Tag the approved spec accept-lock.
   Never edit a check after the tag.
```

COPY THIS

10 rules for your AGENTS.md (2 of 2)

```
6) Report "N of N checks pass".
   Unmet outcome: ABANDON, not silence.
7) Final review gets the diff and the
   checks, never your summary.
8) Max 2 fix rounds. Same finding
   twice: stop and ask. No run over
   40 subagents unless asked.
9) Any price or version claim has a
   date, or the word "unverified".
10) Library question: read current
    docs. Never answer from memory.
```

Install the baseline

```
npx agentic-setup
```

github.com/prayushkale/agentic-setup

Follow Prayush Kale for the next round of trials.

- linkedin.com/in/prayushkale
- x.com/prayushkale
- agentic-playbook.prayush.dev