halvic labs

Context Rule of Thumb: The Blast-Radius Cheatsheet

CHEATSHEET · VERSION: JULY 2026

The rule of thumb

> **Give the agent as much context as the change can do damage.** Context follows blast radius — not the size of the window.

The tiers: change → context

Change	What you hand over
Typo, single-file fix	The file. Done.
Feature within one module	The module + its tests + the interfaces it touches
Cross-cutting change (auth, logging, data model)	The affected subsystems + architecture docs + one or two reference implementations of the convention
Architecture decision or review	As much real codebase as fits, plus ADRs — this is where the million pays off

Side effect: the rule forces you to determine the blast radius *before* the prompt — and that consideration is half the specification.

When big context wins

- **Hand over subsystems, not files** — the agent sees the relationships instead of guessing them; the "it compiles but doesn't fit the architecture" failures shrink.
- **Architecture review over the real codebase** — "where do we violate our layers?" goes from sampling game to honest question.
- **Migration work with memory** — the agent knows conventions and edge cases from the entire diff so far, instead of reinventing them per file.
- **Documents and code together** — spec, ADRs, ticket, and code in one context; the translation losses shrink measurably.

But: retrieval isn't dead. For lookups, targeted search stays cheaper and often more precise — big context wins where *relationships* are the task.

The 3 counterforces

- **Dilution** — relevance is not a function of volume: dump the repo in indiscriminately and the twenty lines that matter become a needle in a haystack.
- **Cost and latency** — large contexts cost money and time linearly, on every call; an 800k context for a one-line change is waste by design.
- **False confidence** — an agent with full context makes *more plausible* mistakes, which are harder to spot; review becomes more important, not less.

The full deepdive: https://halvic.ch/en/blog/1m-token-context-workflow

---