TECHNICAL PAPER · D · WORKING PAPER

REWORK Baselines by Project Type: a 136-Codebase, 16-Year Corpus

The calibration study Paper C calls for. Human in-place editing shares, measured across 1,437 repository-year windows of the largest publicly accessible codebases — so that a low REWORK value can be read against a baseline rather than in a vacuum.

Working paper · first published 14 July 2026 · figures reproduce from public history

ABSTRACT

Paper C introduced **REWORK** — the fraction of a codebase's churn spent editing statements that already existed (CHG / CRN) — and showed it separates hand-maintained from intensively agent-built development. A single gradient across nine codebases is not a scale. This paper supplies the calibration: **1,437 repository-year measurements across 136 of the largest publicly accessible codebases, 2010–July 2026, 393 million logical statements of churn measured**, every figure reproducible from public commit hashes with one command. The finding is a stratified baseline, not a single number. Established mid-size projects (curl, PostgreSQL, an established JavaScript framework) hold their in-place editing share near **17%** — roughly one statement in six — across growth and maintenance alike. The very largest projects (WebKit, the Linux kernel, FreeBSD, Chromium) run lower, **5–10%**, a stratum effect: more vendored and generated content, more contributors landing new code, more growth surface. Intensively agent-built codebases fall **below 1%**. A low REWORK value is therefore evidence only against the baseline for a project of that size and phase — which is what this corpus provides. The instrument is deterministic diff arithmetic; no machine learning enters the measurement.

Ask CodeDelta

IN PLAIN TERMS

> Paper C found that AI-built code is barely edited after it is written — it is replaced, not revised. But to say a number is *low* you need to know what normal looks like. So we measured normal: sixteen years of change history across 136 of the world's biggest open codebases. Ordinary mid-size projects spend about one change in six editing existing lines, decade after decade. The giants spend less — five to ten percent — because so much of their bulk is generated or vendored. Agent-built code spends almost none. Now a single project's figure can be read against the right yardstick.

01 · The question

Paper C's result — established hand-maintained projects rework about one statement in six, an intensively agent-built codebase about one in five hundred — is a gradient drawn through nine points. The open question it names explicitly is calibration: *does the human editing share depend on how old a project is, how large, or what phase of life it is in?* Without that, a low REWORK reading cannot be interpreted — it might mean agent involvement, or it might simply be what a project of that kind always looks like. This paper measures the baselines.

02 · The corpus

QUANTITY	MEASURED
Repository-year measurements	1,437
Distinct codebases	136
Logical statements of churn measured	393,120,776
Span	2010 → Jul 2026

Each row is a calendar-year window (1 January to 1 January; 2026 runs to 2 July) of one repository, measured with the same command that produced every figure in Paper C: `codedelta --git <old>..<new>`. The engine classifies every logical statement as changed in place (CHG), deleted (DEL) or added (ADD); REWORK is CHG / CRN. No sampling. No inference. Every window re-runs from its two public commit hashes.

03 · The strata

The baseline is not one number but three bands, by project size and maturity.

STRATUM	CHARACTER	REWORK BAND
Established, mid-size	Decade-old, hand-maintained (curl, PostgreSQL, an est.-2010 JS framework)	~17%
Mega-project	Multi-million-line, many-contributor, much vendored/generated code	5–10%
Intensively agent-built	Written largely by AI coding agents	<1%

The established band is strikingly stable: it holds near 17% whether a project is growing three-to-one or shedding code, across growth years and maintenance years alike. The mega-project band sits below it — the same measured 2024 windows, one command each:

CODEBASE	WINDOW	CHURN (CRN LLOC)	REWORK
Chromium	2024	4,534,768	9.1%
WebKit	2024	4,973,778	6.0%
Linux kernel	2024	1,436,681	9.3%
FreeBSD	2024	955,664	8.2%

This is a stratum finding, not a contradiction of Paper C: scale itself depresses the editing share, because so much of a mega-project's bulk is new, vendored, or generated code that is added rather than revised. The 17% band was always scoped to established mid-size projects; Paper D reports baselines *by* stratum precisely so that neither is read as the other.

04 · What the baseline is for

The point of a baseline is interpretation. A project measured at 3% REWORK means one thing if it is a mid-size established library (well below its ~17% band — a signal worth investigating) and something entirely ordinary if it is a five-million-line browser engine (squarely in the mega-

project band). REWORK indicates a *development process*, never the authorship of any single change; the failure modes from Paper C hold here too — bulk file moves, vendoring, and greenfield growth all raise the replacement share for reasons that have nothing to do with AI. The corpus does not turn REWORK into a verdict. It turns a number without context into a number read against the right yardstick.

05 · Limits

- **Baselines, not thresholds.** The bands describe populations; no single value is a boundary between assisted and unassisted development.
- **Calendar windows include everything** — vendored rolls, generated files, mass renames. That is deliberate: the corpus measures development as it lands, and the classification lets a reader separate growth from editing.
- **Strata are coarse.** “Established mid-size” and “mega-project” are broad; finer calibration by age and lifecycle phase is ongoing work.
- **A low value is necessary, not sufficient.** Rapid human greenfield growth also collapses the editing share; a low REWORK reading is a pointer for review, not proof of agent involvement.

06 · Reproducibility

Every public window re-runs from its two commit hashes with a single command. The corpus is the instrument of Paper C and is validated in Paper E. The measurement engine is **CodeDelta** (codedelta.app).