Baseline and Improve: Valuing a Change in Tooling

Value Calculators · https://docs.functionize.com/hc/en-us/articles/43795448289559-Baseline-and-Improve-Valuing-a-Change-in-Tooling

For a team that already automates. Rather than comparing claims, this calculator asks you to **measure**: five numbers on what you run today, the same journeys on Functionize for a fixed period, the same five numbers again, and the difference valued in money.

functionize **Studio**
Baseline and improve
www.functionize.com

THE METHOD, IN ORDER

Step 1

Design the comparison, in writing

Same journeys, same environment, same period, same level of familiarity — agreed **before** anything is measured. This is most of the work and all of the credibility.

Step 2

Measure what you have today

Four weeks minimum, six if you can. Write the date on it.

Step 3

Run the same journeys on Functionize

Same length of time. Same five numbers.

Step 4

Value the difference

Maintenance saved, flake avoided, faster feedback, defect cost avoided.

No vendor figure is an input. That is the whole design. Every number in the result came from your own two measurements, which is why it survives a finance review in a way a comparison of claims does not.

WHAT THE DEFAULTS PRODUCE

$60,800	6.7
net annual benefit	months to payback

Those are placeholders until you replace them with your own two measurements. The model is the deliverable; the numbers are yours.

Measure, change, measure again - and value only the difference

www.functionize.com

Measure, change, measure again — and value only the difference.

No vendor figure is an input. That is the whole design, and it is what makes the result survive a finance review.

Download the calculator

Free to download, edit and put your own name on. Every assumption is a cell you can change — nothing is hidden in a formula. **Prefer Google Sheets?** Download the workbook, then in Sheets choose *File → Import → Upload*. Every formula is written to survive the import.

Excel workbook (.xlsx) · 79 KB **How to use it (PDF) · 115 KB**

The method

1. **Design the comparison.** Same journeys, same environment, same period, same level of familiarity — agreed in writing before anything is measured. This is most of the work.
2. **Measure what you have today.** Over four weeks minimum, six if you can. Write the date on it.
3. **Run the same journeys on Functionize** for the same length of time, and measure the same five numbers.
4. **Value the difference** and add the one-time cost of converting the rest of the suite.

The five numbers

Maintenance share	Of the effort spent on automation, how much goes on repairing existing tests rather than adding coverage. Usually the most damning number a team has, and usually the largest single source of value.
Flake rate	The share of results that are non-deterministic for an unchanged version. Count reruns; do not quietly exclude them.
Time to feedback	From a change landing to knowing whether it broke something.
Escaped defects	Per quarter, from your incident or ticket system.
Critical journey coverage	The share of business-critical journeys with a reliable automated check.

These are the same five as What to Measure: QE Metrics and KPIs with Studio, deliberately. Baselining them for a business case and reporting them monthly from then on should be one activity, not two.

Designing a comparison somebody will believe
Same journeys on both sides, and real critical ones rather than the easiest. Same environment, or write down why not. Same level of familiarity — a pilot run by the vendor against a baseline run by a junior engineer is not a comparison. And agree what would make you say no *before* you start: a pilot with no failure condition is a purchase with extra steps.

How the money is worked out

Maintenance saving	Automation effort cost multiplied by the change in maintenance share. *Hard* — both inputs measured.
Flake avoided	Fewer flaky runs, multiplied by what it takes to triage one. *Hard* — both inputs measured.
Faster feedback	Waiting time removed, counted once per release for one engineer. *Soft*, and deliberately conservative.
Escaped defects avoided	The measured change, multiplied by your cost of a defect. *Soft*, because that cost is an estimate.

The sensitivity sheet shows the case with both soft numbers removed. If it holds on maintenance and flake alone — both of which come straight from your own run history — the case is about as solid as these get.

Two things people get wrong

- **Baselining after the pilot has started.** The single most common way these exercises become unusable. Measure first.
- **Guessing the conversion time.** Convert ten tests during the pilot and use the real figure. It is the input people most often estimate, and most often estimate low.

What the platform costs you

You do not type in a license figure. The **Platform and AI cost** sheet works it out from your plan tier and what you expect to consume, and the business case reads the answer from there.

Three things it asks for. Your **plan** — monthly fee, credits included, and the price of an additional credit block, all as input cells so they can be checked against current pricing or replaced with the figures on an Enterprise order form. Your **consumption** — tests built per month, runs per month, and repairs per month, priced at the published illustrative rates of 200,000 credits for a new mid-length test and 4,000 for a run. And, optionally, your **LLM spend**, if you drive Studio from Claude Code, Cursor or another tool over MCP.

Two things worth knowing before you present it. The MCP block is **additional, not a substitute** — driving Studio from your own AI tool does not reduce credit consumption, because the same agent work and the same test runs still

happen. And **month one is not the steady state**: building a suite from nothing costs far more credits than running it afterwards, so the first invoice will not look like the twelfth. Say that to whoever approves the spend before they see it, not after.

If the annual figure comes out small next to the labor saving, that is the published pricing working as intended rather than an error — but it does mean the case now rests almost entirely on your own effort and defect numbers. Take the sensitivity sheet into the review, not this one. There is an override cell for Enterprise customers with a negotiated annual figure.

See How Credits Work for where the credit rates come from.

Where Studio fits

Three of the five numbers come out of Studio directly once the pilot is running: Reports: Pass Rates, Trends and What They Actually Tell You for pass rate and warnings, orchestration run history for duration, and the History panel for what changed and when. See Orchestrations: Running a Suite as One Job.

Replacing manual effort rather than comparing tooling? Use Manual to Automated: What the Change Is Worth instead.

Related

- Value Calculators: Building the Business Case — the other calculator, and how both are built
- What to Measure: QE Metrics and KPIs with Studio — the same five numbers, reported monthly
- Keeping a Suite Green: Flake, Tiers and Who Owns Red — flake and maintenance, which carry most of the value
- Reports: Pass Rates, Trends and What They Actually Tell You — where three of the five numbers come from