How to Use the Baseline and Improve Calculator

Valuing a change in tooling from evidence you gathered yourself

This calculator is for a team that already automates. Rather than comparing claims, it asks you to measure five numbers on what you run today, run the same journeys on Functionize for a fixed period, measure the same five again, and value the difference. No vendor figure is used as an input, which is what makes the result survive a finance review.

> ***How to use this template.*** *Read section 2 before you measure anything. The pilot design decides whether the result means anything at all. Measure for at least four weeks on each side. Six is better. If you did not measure something, leave the baseline value in place rather than estimating. The model will show no benefit for it, which is the honest answer. Convert ten tests during the pilot and use the real conversion time. It is the input people most often guess, and most often guess low.*

1. The method, in four steps

1. **Design the comparison.** Same journeys, same environment, same period, same level of familiarity. Agree it in writing first.
2. **Measure what you have today.** Five numbers, over a representative period, with the date on them.
3. **Run the same journeys on Functionize** for the same length of time, and measure the same five.
4. **Value the difference** and add the one-time cost of converting the rest of the suite.

> The order matters. Baselining after the pilot has started is the single most common way these exercises become unusable.

2. Designing a comparison somebody will believe

This is most of the work, and it is worth being pedantic about. An unfair comparison produces a number nobody believes - including you, six months later.

- **Same journeys on both sides.** Real critical journeys, not the easiest ones. Ten is usually enough.
- **Same environment**, or write down why not and what it does to the result.
- **Same level of familiarity.** A pilot run by the vendor against a baseline run by a junior engineer is not a comparison.
- **Count reruns on both sides**, and say so. Retry policy moves pass rate more than most tooling differences do.
- **Agree what would make you say no**, before you start. A pilot with no failure condition is a purchase with extra steps.

3. The five numbers

Number	Where it comes from
Maintenance share	Of the effort spent on automation, the share that goes on repairing existing tests rather than adding coverage. Usually the most damning number a team has, and usually the largest 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, over a full quarter.
Critical journey coverage	The share of business-critical journeys with a reliable automated check. The one an executive can interpret.

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

4. How the money is calculated

Benefit	How it is valued	How soft it is
Maintenance saving	Automation effort cost multiplied by the change in maintenance share.	Hard. Both inputs are measured.
Flake avoided	Fewer flaky runs, multiplied by the time it takes to triage one.	Hard. Both inputs are measured.
Faster feedback	Waiting time removed, counted once per release for one engineer.	Soft. Deliberately conservative, and easy to strip out.
Escaped defects avoided	The measured change, multiplied by your cost of a defect.	Soft, because the cost of a defect 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.

5. The three challenges, and the answers

- **"The pilot was not representative."** Answer it in advance with the pilot design sheet: same journeys, same environment, same period, agreed before you started.
- **"You are counting engineer waiting time twice."** The model counts it once per release, for one engineer. Say so, and offer the version with it stripped out.
- **"Conversion will take longer than that."** Likely, if the estimate was not measured. This is why you convert ten tests during the pilot.

6. For the customer success conversation

The job here is not to show a number. It is to help a team design a comparison they will trust, and then to stay out of the way of the measurement.

1. Run the pilot design session first, as its own conversation. Get the rules written down and agreed.
2. Make sure the baseline is measured by the customer, not by us. A baseline we produced is the first thing a reviewer discounts.
3. Resist the urge to pick favourable journeys. The credibility of the whole exercise rests on this.
4. When the measured numbers come back, fill the sheet in together and let the result be what it is.
5. If a number went the wrong way, say so and put it in the model. A business case with one honest negative is far more persuasive than one where everything improved.

> A measured 18% improvement on the customer's own suite is worth more in a review than a claimed 70% on somebody else's. Keep the conversation on their evidence.

7. After the decision

Keep measuring the same five numbers monthly. The business case then becomes the first data point in an ongoing series rather than a document that gets filed.

Date of the baseline measurement

Date the pilot finished

Which number moved least, and why

Date of the first monthly review

> **With Functionize Studio:** Reports gives you pass rate and trends across projects, orchestration run history gives you duration for time to feedback, and the History panel records every change with its author. See *What to Measure: QE Metrics and KPIs with Studio*.