SG1 CONSULTING

WHITE PAPER

ROI & BASELINE

Measuring Automation ROI: Baseline the Rework First

A measurement discipline for regulated-sector evaluators: capture the true cost of your current process before you orchestrate anything, so the return is proven rather than asserted.

For compliance, procurement, legal-ops, and process owners · Updated August 2026

The direct answer: measure the rework, not the promise

A credible automation ROI is a subtraction between two numbers measured the *same* way: what the process costs today, minus what it costs after. If you never measured "today," the "after" has nothing to subtract from — and the return is an opinion dressed as a figure.

Most automation business cases fail their own audit because the "before" was never captured. Savings get back-calculated from a vendor's assumptions instead of the buyer's own baseline, and the most common single error is the crudest one: dividing tool cost by the headcount it supposedly displaces. That calculation ignores rework, quality, and exception costs, and it misstates the labor figure — usually by a wide margin, because it prices people at wage rather than at what they actually cost to employ.

Rework is where automation actually pays. The hours spent finding and fixing mistakes in the current process are a hidden tax that rarely appears on any ledger, yet they are the most reliably recoverable cost you have. This paper reverses the usual order of operations: it names the measurements to take on the manual process

before any orchestration is switched on, so post-automation ROI becomes a defensible subtraction rather than a claim you have to trust. For a regulated buyer there is a second payoff — a baseline is also your audit defense. It converts "we believe this helped" into a documented before/after a reviewer can re-derive.

The three moves everything else operationalizes

- **Freeze a baseline window** before you sign anything — a representative stretch of the real process, not one clean day.
- **Define each metric once, in writing,** before the window opens — a definition that drifts mid-stream silently voids the comparison.
- **Price time at fully-loaded cost,** not wage — rework valued properly is the number that moves the model.

1. What poor-quality cost really is — and why it hides

Quality management has a durable vocabulary for this, and it is worth borrowing because it stops you from double-counting or over-claiming. Costs split into four buckets. Automating a document or approval process attacks mainly two of them: **internal failure** and **appraisal**. Naming the bucket each claimed saving lives in is the discipline that keeps an ROI honest.

Bucket	What it is	Does automating an approval/document process touch it?
Prevention	Effort spent stopping defects before they happen — training, better forms, design of the control itself.	Indirectly. Good automation bakes prevention in, but don't claim it as cash.
Appraisal	Checking and inspection to catch defects — manual review, second sign-off, reconciliation.	**Yes.** Much of this can shrink — but keep the checks a regulator requires.
Internal failure	The cost of defects caught before they leave — rework, re-runs, re-checks, send-backs.	**Yes — the primary target.** This is the recoverable prize.
External failure	Defects that escape — remediation, penalties, findings, lost trust.	Only if the automation itself stays safe. A bad automation *adds* to this bucket.

Poor-quality cost commonly consumes a meaningful share of the cost of running a process, and in low-maturity operations it is far larger than leadership assumes. The catch: the visible portion — the obvious redo work — is only a fraction of it. The firefighting, the escalations, the schedule slip, and the relationship damage typically run several times the documented figure.

The hidden factory

There is a parallel operation inside most processes that exists only to catch and fix defects: the second reviewer, the reconciliation step, the "just double-checking" email, the item that gets quietly sent back and redone. It is real labor that no org chart shows, because rework is logged as normal work rather than as failure. It never rolls up into a number anyone reports. If you do not instrument it deliberately during the baseline, it stays invisible — and your ROI understates the real gain.

> **Recoverable vs. irreducible — the line that keeps you honest**
>
> Separate the poor-quality cost automation can *remove* (the redo work) from the cost you must *keep* (the checking a regulator or your own risk appetite requires). Only the recoverable portion belongs in a payback claim. Counting a mandated control as a "saving" is the fastest way to have your business case fall apart under review.

2. Draw the process before you time it

The classic baselining mistake is to measure an idealized process — the one in the procedure document — instead of the one that actually runs. Map the real path first, including the unofficial steps: the rework loops, the reassignments, the "send it back" arrows. The exceptions are usually where both the cost and the automation opportunity live, so a map that omits them will baseline away the very thing you are trying to value.

As you map, do four things that later determine whether your savings are legitimate:

While mapping	What to capture	Why it changes the ROI
Tag every step	Value-add · required-but-non-value-add (a mandated control you must keep) · pure rework/waste.	Only waste and avoidable rework can be claimed as savings. The tagging is the audit trail for that claim.

While mapping	What to capture	Why it changes the ROI
Mark handoffs & waits	Every queue and every wait, separately from hands-on work.	Wait time is often the majority of elapsed cycle time and is invisible in a task-duration-only view.
Flag the irreversible points	Steps where something can't be taken back — a payment released, a filing submitted, a record deleted.	These must stay human-confirmed after automation, so they shape both the design and the honest savings estimate.
Capture volume & variation	How often each path runs, not just the happy path.	A process that is clean most of the time and chaotic in a minority of cases earns its ROI from the chaotic minority.

3. The baseline measurements to take before orchestration

This is the core of the discipline. A baseline is defensible only if it is representative, defined in advance, and priced correctly. Take these measurements on the live manual process before a single automated step is switched on.

Capture a representative window

Measure across a full business cycle — ideally several months where the historical data exists — spanning different people, different days, and different load conditions. One clean sample on a quiet day is not a baseline; it is the number the process hits when nothing goes wrong, which is precisely the condition automation is *not* being bought to address.

The core metric set — each defined once, in writing

Metric	Definition to lock before the window opens
End-to-end cycle time	Elapsed time from the moment work arrives to the moment it is fully done — including every wait and queue, not just active work.
Hands-on touch time	Actual working minutes a person spends on the item. Distinct from cycle time; conflating the two hides the wait.
First-pass yield	Share of items completed correctly the first time with no rework, send-back, or re-check.

Metric	Definition to lock before the window opens
Error rate	Share of items with a defect, however caught. Define what counts as an error before you start counting.
Exception / escalation rate	Share of items that leave the standard path — escalated, reassigned, or handled by hand.
Rework hours	Time spent finding and fixing defects. Instrumented explicitly (see below), not inferred.
Volume per period	Items processed per week or month, by path, so savings scale to real throughput.

Price time at fully-loaded cost

Value rework and touch time at the fully-loaded cost of the people doing it — salary plus benefits plus overhead — which is a substantial uplift over base pay. Rework priced at loaded cost is the figure that actually moves an ROI model; rework priced at wage quietly understates the recoverable prize.

Instrument rework explicitly

Your normal systems record redo work as ordinary work, so rework will not surface on its own. During the baseline window, add a simple reason code or tally for every "sent back / redone / re-checked" event. This one deliberate instrument is what converts the hidden factory into a measured, defensible number.

> **Two collection modes — triangulate, don't trust one**
>
> **System and log extraction** is objective but misses off-system work — the phone calls, the side spreadsheets, the corridor fixes. **Observation and self-report** captures that hidden factory but is disruptive and can bias behavior while people know they are watched. Use both and reconcile them; where they disagree, the gap is itself a finding. Write the whole measurement plan and every metric definition down before the window opens — a definition changed mid-stream silently invalidates the before/after comparison you are building the entire case on.

4. An honest ROI model

With a real baseline, the model builds itself — and the honesty is in the construction, not in a disclaimer at the end. Keep the numerator fully loaded, build the savings side

only from metrics you actually measured, and never assume a step disappears entirely when a human still has to load, monitor, or confirm it.

Model element	How to build it honestly
Payback period	Automation investment ÷ measured annual saving. Put *everything* in the numerator: build, license, change management, and the human-confirmation time that remains after automation. Don't hide implementation labor.
Savings side	Sum only what the baseline measured: avoided rework hours, reduced cycle time, lower exception rate, and reduced manual-review effort — each traced back to a specific metric.
Fractional displacement	If a human still loads, monitors, or confirms the step, count fractional capacity freed, not a whole role. Claim the conservative figure and let real results beat the projection.
Hard vs. soft savings	Put cashable, verifiable items (removed rework, avoided remediation) in the headline. Keep "freed-up time" as a clearly labeled secondary line — never the whole case.
One-time vs. recurring	Separate them. A one-time cleanup is not an annual saving, and treating it as one inflates every year after the first.
Risk & compliance line	The one regulated buyers answer to: reduction in control failures, audit findings, and remediation events — measured on the same before/after basis, not asserted.

Present three scenarios, lead with the conservative one

Model conservative, expected, and optimistic cases — and make the decision on the conservative one. If the conservative case already clears your payback threshold, the decision is robust to being wrong, which is exactly the property you want to defend to a board or an auditor. If only the optimistic case clears the bar, you have found out something important before signing rather than after.

5. Design the pilot so the number is trustworthy

A pilot exists to isolate the automation's effect from everything else that might move the metrics — a staffing change, a quiet month, a motivated team on its best behavior. State its job narrowly and in advance: prove *this* process, in *this* environment, on *this* data, clears a pre-set bar. Write the pass/fail threshold before you start, not after you see the results.

Pilot discipline	What it prevents
Matched comparison	Run automated and non-automated volume side by side, or the same team before and after on equivalent work. Prevents attributing to automation a gain that came from a staffing or volume change.
Aligned measurement periods	Measure the identical metric set across four stages — baseline, pilot, ramp, steady-state — using the same definitions. Prevents a comparison that quietly changed its own yardstick.
Automation-specific metrics	Track straight-through rate (items handled end-to-end with no human touch), error-rate delta on equivalent volume, exception/fallback rate, and rework the automation itself introduces.
Instrument the failure modes	Record what the automation got wrong, how often it stopped and asked, and how often a human had to unwind an action. These decide whether the ROI survives contact with production.
Predefined kill/scale criteria	Set the thresholds that end or expand the pilot in advance, so it can't be talked into a "success" it didn't earn.

> **The plainest statement of the whole discipline**
>
> Without a pre-automation baseline, every claim about the pilot is an opinion. The baseline is not paperwork you do to satisfy procurement — it is the only thing that lets you say, with evidence, that the automation and not something else produced the result.

6. How a well-built platform keeps the savings safe

An ROI only counts if the automation cannot manufacture new failure cost — the expensive external-failure bucket — the moment you switch it on. Six platform properties are what keep the gain from being erased by a new remediation line item. Ask a vendor to *demonstrate* each on a real action, not describe it in principle.

Undo by design

Every reversible automated action ships with a defined, tested reversal built alongside it, so a way back always exists — and an error stays an error instead of becoming an irreversible loss that wipes out the return.

Make them show it: pick one action and ask them to reverse it live.

Confirm before commit

Consequential or irreversible actions do not run unattended; they stop and wait for a named person — the same human-confirmation steps you flagged when you mapped the process.

Make them show it: ask which action classes are gated, and who can confirm each.

No self-approval

The system cannot fabricate its own sign-off. A real human act is required, so the automation can never invent an approval to push a step along.

Make them show it: ask how the platform proves an approval came from a person, not the automation.

A record that can't be rewritten

Every action and every reversal is written to a log that no one — including the vendor — can edit or delete. This is the permanent record that makes your before/after ROI and your control posture re-derivable by an auditor.

Make them show it: ask who *can* alter the log. The right answer is "no one."

Stops when unsure

When a permission, check, or version doesn't line up, the step halts and escalates rather than committing something it can't take back.

Make them show it: ask what happens on ambiguity. The safe default is "stop," not "proceed."

Least access per task

Each function runs with only the permissions it needs, on separate credentials, so a fault or leak in one workload cannot reach another — containing the blast radius, and the remediation cost that would otherwise erase the gain.

Make them show it: ask whether one compromised component exposes everything, or only its own slice.

The tie-back to ROI: these six properties are what keep external-failure cost — the expensive bucket — from reappearing as a new line the moment you automate. A saving that arrives with a fresh remediation liability attached is not a saving.

7. The baseline worksheet

Fill one of these per process before automating. The left column is what you measure; the middle is where you record it; the right shows how the figures roll into an annual poor-quality cost and a conservative payback. The example values below are illustrative placeholders to show the arithmetic — replace every one with your own measured numbers.

Line	Your measured value	How it rolls up
Volume per period	e.g. items / month	Scales every per-item figure to real throughput.
End-to-end cycle time	e.g. hours / days per item, incl. wait	Before/after delta = cycle-time saving.
Hands-on touch time	e.g. minutes per item	× loaded rate = labor cost per item.
First-pass yield	e.g. % done right first time	The complement is your rework population.
Error / exception rate	e.g. % defective or escalated	Drives appraisal and internal-failure cost.
Rework hours per period	from the reason-code tally	× loaded rate = recoverable rework cost.
Fully-loaded labor rate	wage + benefits + overhead	The multiplier applied to all time figures.
= Annual poor-quality cost	(rework + avoidable appraisal) × periods	The baseline the saving subtracts from.
= Conservative annual saving	recoverable portion only, fractional FTE	Numerator’s partner in the payback ratio.
= Payback period	total loaded investment ÷ conservative saving	Decide on this figure, not the optimistic one.

8. The vendor scorecard this worksheet feeds

The same baseline exposes the vendor claims that don’t hold up. Score each row on *evidence*: **0** if it is only asserted, **1** if documented, **2** if demonstrated. Weight by what your workflow actually needs.

Question to put to the vendor	Weight	Score
Will you build the ROI against OUR measured baseline, not your benchmarks?	**High**	0 / 1 / 2
Is there a defined plan to measure the “after” on the same definitions?	**High**	0 / 1 / 2
Do irreversible actions require a named human to confirm?	**High**	0 / 1 / 2
Is there a tested, per-action undo you can demonstrate?	**High**	0 / 1 / 2
Can the system approve its own actions? (The right answer is no.)	**High**	0 / 1 / 2
Is the activity log append-only and tamper-evident?	**High**	0 / 1 / 2
What happens when a check, permission, or version fails to line up?	**Med**	0 / 1 / 2
How are permissions scoped per workload, and what’s the blast radius?	**Med**	0 / 1 / 2
Does the model assume full headcount elimination, or fractional?	**Med**	0 / 1 / 2
Are “soft” savings kept separate from cashable ones?	**Med**	0 / 1 / 2

Treat any **0 on a High-weight row as a stop, not a deduction.** A vendor can post a strong total and still be disqualified by a single high-weight zero — a business case with no plan to measure the after, or an automation that can rewrite its own record, isn’t one you can defend later.

The data traps that quietly break a baseline

- Sampling only clean, low-volume days, so the baseline understates the rework automation would remove.
- Counting touch time as cycle time, hiding the queue and wait that are often the majority of elapsed time.
- Pricing labor at wage rather than fully-loaded cost, understating recoverable rework.
- Letting a metric definition drift between the baseline and pilot windows, voiding the subtraction.
- Accepting a vendor ROI built on their benchmarks, with no plan to measure your own “after.”

- Automating the idealized process map instead of the real one, so the exception loops where the cost lives are never measured.

How to run it

Map the real process, including its rework loops. Freeze a representative window and write down every metric definition before it opens. Instrument rework with a reason code, and price all time at fully-loaded cost. Build the model from what you measured, lead with the conservative case, and put the risk-and-compliance reduction on the same before/after footing as the labor saving. Run a matched pilot against a pre-set threshold. Circulate one document — the baseline window, the metric definitions, the pilot thresholds — that procurement, compliance, and the process owner all sign. One set of numbers, measured the same way twice, is what turns a return from a claim into a fact. Confirm any regulatory obligation touched by the process with your own counsel or authorizing official.

SG1 CONSULTING

sg1consulting.us · contact@sg1consulting.us