Lab Automation: Which Processes Pay, and Which Ones Do Not

A category built on optimism deserves a sceptical test. Three conditions decide whether a process is worth automating, and knowing which of yours qualify is the whole exercise.

Brandon Holland, Marketing & Operations Digital Solutions Architect

July 6, 2026 · 8 min read

IN BRIEF

Lab automation pays where a task is high volume, low judgement and methodologically stable. Where any one of those three is missing it usually costs more than it returns, because the exceptions consume the saving. Most laboratory work fails at least one condition, which is why candidate selection matters more than technology choice.

Key takeaways

- Three conditions decide it: high volume, low judgement, stable method. All three together, not two of three.
- The saving is destroyed by exception handling, so count the exceptions before you count the volume.
- Transcription is the cheapest automation win and rarely called automation, because the fix is an interface, not a robot.
- A method that changes more than once a year is a poor candidate at any volume, because each change is a revalidation.
- If nobody can describe the current process on one page, automating it encodes the confusion and then hides it.

Every lab has a list of things it would automate given the budget. Most of those lists are ordered by how annoying the task is, which is a reasonable instinct and a poor predictor of whether automating it will pay.

Laboratory automation, in the plain sense, is any step where a machine or a rule does what a person used to do: moving a sample, reading an instrument, deciding who reviews a result. The definition is not the hard part. The hard part is that automation pays where three conditions hold together, and only together. **A qualifying process is high volume, low judgement, and runs a stable method. All three, not two.** Miss any one and the saving is consumed by exceptions a person still handles. Most laboratory work fails at least one, which is why choosing the candidate matters more than choosing the technology.

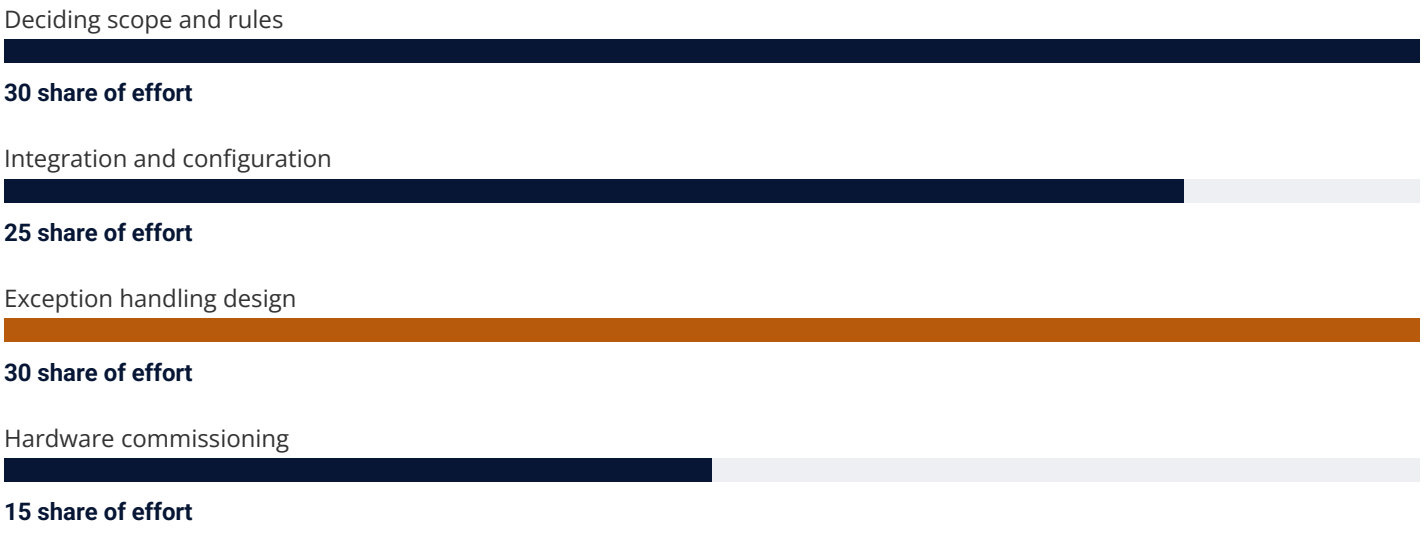
What makes a process worth automating?

Volume. High enough that a per-run saving accumulates into something visible. This is the condition labs assess correctly, and often the only one.

Low judgement. The decision can be written as a rule. If an experienced analyst decides on the basis of experience, that step becomes an exception queue with a person at the end of it.

Method stability. The configuration will not be rewritten. A method changing more than about once a year is a poor candidate at any volume, because each change is a reconfiguration and, in a regulated lab, a revalidation.

WHERE EFFORT LANDS ON A FIRST AUTOMATION PROJECT



Composite of the phases described here

The third bar is the one to look at. Exception handling is about as large as the configuration and gets estimated last, usually once the happy path already works.

Why does exception handling destroy the business case?

Because the saving is calculated on the happy path and the cost is incurred on everything else.

EXCEPTION RATE	WHAT THE CASE USUALLY CLAIMS	WHAT ACTUALLY HAPPENS
Under 2%	Fully automated	Broadly true; the residue is absorbed
5% to 10%	90 to 95% automated	A second, more skilled manual path now exists alongside the first
Over 15%	Automated with edge cases	Two processes, and staff must first work out which one a sample is in

Counting exceptions before counting volume is the more useful order. A process with a ten percent exception rate and no exception design is not ninety percent automated. It has a new failure mode, and the people handling it now need to understand both the old process and the new one.

What does lab automation look like in practice?

The word covers a robot arm and a checkbox with equal comfort, which is part of why business cases go wrong. These are the categories that come up in real labs, with the condition that decides whether each one pays.

EXAMPLE	WHAT IT REPLACES	QUALIFIES WHEN
Liquid handling and sample preparation	A technician pipetting, diluting, aliquoting	Volume is high and the prep protocol has not changed in a year
Instrument data capture	Reading a number off one screen and typing it into another	Almost always; the fix is an interface, not a robot
Review routing	Deciding by convention who looks at which result	The routing rule can be written down and agreed
Barcode-driven sample tracking	Handwritten labels and a logbook	Any lab with more than one person handling the same sample
Inventory and calibration scheduling	A spreadsheet somebody remembers to check	Expiry and calibration dates are already recorded somewhere
Robotic handling in manufacturing QC	Loading, unloading and transporting samples between stations	Throughput is genuinely limited by physical handling, not by review

Two of those six need hardware. The other four are software: an interface, a rule, a barcode, a schedule. Labs shopping for automation tend to picture the first row and buy toward it, while the largest saving on the list is usually sitting in the second. In a manufacturing QC lab the robotic row can be real, because the constraint really is physical; in most testing labs it is not, and the handler waits on the reviewer.

How do you find the candidates in your own workflow?

By counting touches, not by asking people what annoys them. Walk one sample through the lab and record every point at which a human moves it, reads it, writes about it, or decides something. Then sort the touches.

- **Moves** need hardware, and are the expensive category.
- **Transcriptions** need an interface, and are almost always the cheapest win.

- **Decisions** need a rule. Where the rule cannot be written, the decision stays with a person, and that is the correct answer.

Most labs doing this find their best first candidate is a transcription they had stopped noticing, not the instrument they were planning to buy a handler for. Sample preparation is usually the second: repetitive, error-prone and judgement-free, and chosen late because the instrument is the expensive object even though instrument time is rarely the real bottleneck. Instrument integration covers the mechanics of the transcription case, and the lab automation hub the wider category.

What does automation change about validation?

It moves the burden from the result to the decision. A manual step is validated by demonstrating that a competent analyst following the procedure produces the right answer. An automated step has to demonstrate that the rule produces the right answer across the range of inputs it will actually see, including the ones nobody thought about. That is a wider evidence requirement, not a narrower one, and it is the part most often underestimated.

- **Boundary conditions.** What the rule does at exactly the threshold, not comfortably either side of it.
- **Failure behaviour.** What happens when an instrument returns nothing, or returns something malformed, rather than something wrong.
- **Change control.** Who can alter the rule, and what revalidation their change triggers.

21 CFR 11.10(a) asks for systems validated to give “consistent intended performance”, and an automated step with no record of what it did cannot demonstrate that. The record is the evidence, which is why the software half of automation matters to the regulator as much as the hardware half matters to the schedule. Our validation services page covers how that evidence is structured.

Where is lab automation heading?

Three directions, none of them the robot-filled room the category was sold on.

Modular over monolithic. The large integrated automation line, designed once for a fixed workflow, is losing ground to benchtop modules that can be rearranged when the method changes. That is a direct response to the third condition above: a lab whose methods change cannot commit to a fixed line, and increasingly does not have to.

Interfaces are becoming standard. Instrument integration has historically meant writing a parser for each vendor's file format. Open standards such as SiLA 2 define how an instrument describes itself and exchanges commands and data, so that connecting a new device becomes configuration rather than a project. Adoption is uneven and older instruments will never speak it, but the direction is clear and it attacks the single largest cost in the chart above.

Software is carrying more of the decision. A laboratory execution system enforces a method step by step, refusing to let an analyst proceed until the previous step is recorded. Rule-based review routing surfaces the exceptions rather than waiting for them. And newer agent-style tooling is beginning to draft the routine decisions for a person to confirm. Each of these is automation without hardware, and each is judged by the same three conditions. The piece on AI agents in the laboratory covers where that last category genuinely helps and where it is still a demo.

Where do these projects actually go wrong?

Scope that grows quietly. Phase one starts as four instruments and is eleven by month three, because each addition was individually reasonable and nobody decided to double the project.

Automating an undescribed process. If the current process cannot be written on one page that two people agree with, automation encodes the disagreement and hides it, and the two sections that disagreed now each believe the system is wrong.

Assuming headcount falls. What changes is usually what people do, moving effort from handling to review and troubleshooting. A case built on redeployment survives contact with reality; one built on reduction invites a reckoning.

Buying the hardware first. The handler arrives, the workflow it was meant to serve has not been written down, and the integration to the system that should receive its output was never scoped. The most expensive object in the project spends its first six months as furniture.

> NOTE
>
> Some processes should stay manual. A low-volume, high-judgement test performed by one experienced analyst is a sensible allocation of a skilled person, not a failure of modernisation.

What a LIMS carries in this work

The unglamorous half, and usually the half that pays first: receiving instrument output directly so transcription disappears, routing review by rule so exceptions surface, and recording what the automation did as auditable events. That last one is what lets an automated step demonstrate the “consistent intended performance” the regulation asks for.

Hardware automates the handling. Software decides what happens when the handling does not go to plan, which is where the difference between a project that pays and one that does not usually sits.

> **How LabLynx handles this**
>
> LabLynx handles the capture, routing and exception surfacing above, which is the software half rather than the hardware half. If your bottleneck is genuinely physical throughput, a LIMS will not fix it and we would point you at instrument vendors first. If it is transcription, review or the exceptions nobody designed for, that is the half we build.

Frequently asked questions

What makes a laboratory process a good automation candidate?

Volume high enough that the per-run saving accumulates, judgement low enough that the decision can be expressed as a rule, and a method stable enough that the configuration will not be rewritten within the year. All three at once. A process that is high volume but changes method every quarter will cost more in reconfiguration and revalidation than it saves in handling.

What are common examples of laboratory automation?

Liquid handling and sample preparation, direct capture of instrument readings, rule-based review routing, barcode-driven sample tracking, inventory and calibration scheduling, and robotic sample handling in manufacturing QC. Only the first and last need hardware. The other four are software, and the instrument-capture case is usually the largest saving available in a testing lab.

Should we automate sample preparation or analysis first?

Usually preparation, and labs usually choose analysis. Preparation is more repetitive, more error-prone and less judgement-dependent, which is precisely the profile that qualifies. Analysis gets chosen first because the instrument is the expensive object in the room, but instrument time is rarely the actual bottleneck; the samples waiting to be prepared for it are.

What is a laboratory execution system?

Software that enforces a method step by step: the analyst cannot proceed to step four until step three is recorded, with the right reagent lot and the right instrument. It is automation of the procedure rather than of the handling, and it qualifies on the same three conditions. It pays where methods are stable and volume is high; it becomes an obstacle where the method is still being developed.

Why do automation projects over-scope?

Because each addition is individually reasonable. A second instrument, three more report templates, one more sample type: none is unreasonable and the total doubles the project without anyone deciding to. The defence is a written phase-one scope with a named owner who has to approve every addition, and a schedule that is revised when they do.

Does automation reduce headcount?

Rarely in the way business cases assume, and claiming it invites a reckoning. What it more often changes is what people spend time on, moving effort from handling samples to reviewing results and troubleshooting exceptions. A case built on redeploying skilled people to that work survives contact with reality; one built on removing them usually does not.

Sources and references

1. 21 CFR 11.10 Controls for closed systems

   Electronic Code of Federal Regulations

2. SiLA 2 Standard for laboratory instrument interoperability

   SiLA Consortium

Brandon Holland

Marketing & Operations Digital Solutions Architect

Brandon Holland is LabLynx's Marketing & Operations Digital Solutions Architect. He builds and runs the digital systems behind the company's public surface, from the website and content architecture to the tooling that keeps product information accurate everywhere it appears. He writes about laboratory informatics from the systems side: how lab records get structured, searched, and kept defensible as a lab scales.

Read the current version online

https://www.lablynx.com/resources/articles/lab-automation-trends/

Downloaded September 24, 2026. LabLynx builds laboratory informatics software: LIMS, ELN, LIS and lab automation.