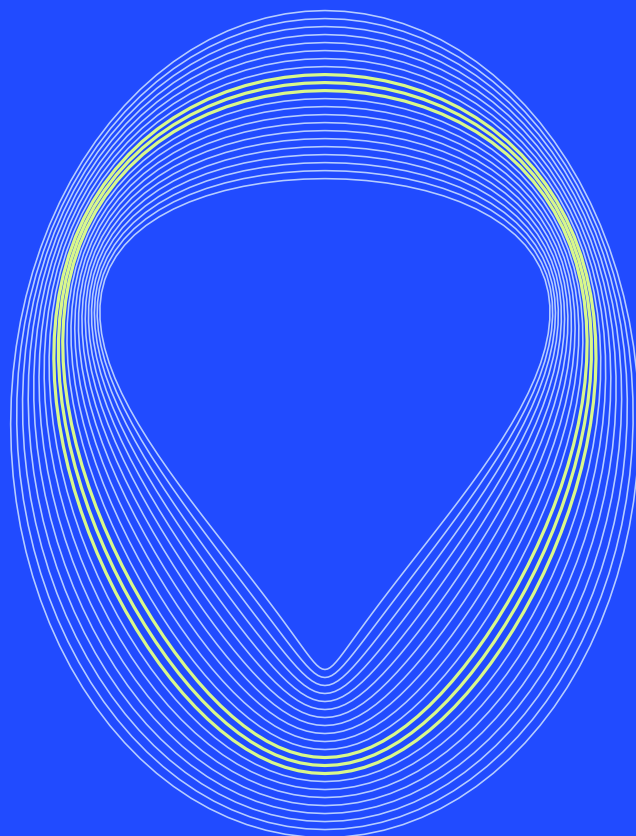


Reinventing the wheel, on purpose.

Why AI makes it worth building software around the business again.



[Executive summary](#)

Software should fit the business.

For years, the default advice was to buy software and adapt the business to it. AI gives us a reason to reopen that decision. Our hypothesis is that lower implementation effort can make purpose-built software viable for more companies and more processes. The opportunity is to preserve what makes a business effective, improve what creates friction and put intelligence directly into the work.

A decision worth reopening.

AI can change implementation effort and expand where purpose-built software is viable.

Two roles for AI.

Accelerate delivery and embed intelligence in the company's operation.

One complete responsibility.

Evaluate fit, quality, operation and maintenance, as well as implementation.

The business is the starting point.

A familiar category does not make every company's requirements the same.

"Don't reinvent the wheel" is useful advice when the wheel already fits. In software, it has also become a shortcut for a harder decision: how much should a company change the way it works to fit the product it can buy?

A ready-made product distributes development cost across many customers. That is a powerful model. But its standard workflows, data structures and release priorities are also designed for many customers. The gap between the product and one company's actual operation often becomes configuration, custom integrations, spreadsheets, duplicate entry and work that happens outside the system.

Impactia's hypothesis is that AI-assisted engineering can change the economics of closing that gap. When implementation requires less effort, a tailored application becomes worth considering in situations where it previously looked too expensive or too slow. The decision should be reopened around the value of fit, not settled by the existence of a software category.

This is a position about where to invest. It is not a prediction that every company will replace its existing systems. The useful question is which parts of an operation deserve software shaped specifically around them.

Build where the fit creates value. Reuse where the problem is already solved.

The economics are moving. The responsibility remains.

Faster code generation is one input into delivery, not a measurement of business value.

AI can assist with implementation, tests, documentation, code exploration and iteration. Our inference is that reducing effort in these activities can expand the range of projects worth attempting. That inference depends on the team, the task, integration demands and the quality standard the software must meet.

Published results are mixed. Across three field experiments involving 4,867 developers, Cui and colleagues reported 26.08% more completed tasks with an AI coding assistant; the pooled estimate had a standard error of 10.3 percentage points. It is a measured result in those settings, not a delivery guarantee. [1]

In a different setting, METR's early-2025 trial with 16 experienced open-source developers found that AI-assisted tasks took 19% longer. Its February 2026 update described selection and measurement problems that made the newer estimates unreliable as a measure of current uplift. [2] [3]

DORA's 2025 research describes AI as amplifying existing organizational strengths and weaknesses. Our reading is that clear requirements, review, testing and feedback remain central to turning tool capability into useful delivery. [4]

The evidence does not establish one universal productivity multiplier, a standard project price or a guaranteed time to production. Our proposition is more practical: use AI deliberately, define a bounded release and measure whether the complete delivery process improves.

Treat speed and cost as outcomes to demonstrate within a defined scope.

Fit the software to the work worth preserving.

Custom does not mean turning every existing habit into a feature.

Every company contains both valuable differences and accumulated friction. A decision rule may reflect years of operating knowledge. A manual handoff may exist only because two systems never connected. They should not receive the same treatment.

Before designing the application, we work with the people who perform the process and the people accountable for its results. We identify the decisions that matter, the information they depend on, where exceptions arise and what a completed outcome actually means. This is the human work that determines what the technology should do.

Then we separate three things: constraints that must be respected, practices that create an advantage and steps that can be removed or redesigned. The goal is a better operation expressed in software. Copying a poor process at higher speed is an expensive way to preserve the problem.

A bespoke system should make the important business rules explicit. It should give each role a useful view, connect to the systems that own the data and let the company change its workflow without waiting for a general-purpose vendor's roadmap. Those benefits require design and ongoing ownership; they do not come from generating code alone.

- Preserve the rules and relationships that make the business effective.
- Simplify avoidable handoffs, duplication and unnecessary approvals.
- Make exceptions visible, with an owner and a clear next action.

The process is the starting material. A better operation is the design objective.

AI in the making. AI in the operation.

The way we build and the capabilities we deliver are two connected decisions.

AI-assisted delivery concerns the engineering process. It helps the team explore, implement and iterate while people remain accountable for requirements, architecture, review and release decisions. Its value is assessed through working software, quality and the effort required to reach an accepted outcome.

Applied AI concerns the application itself. It can interpret unstructured information, retrieve relevant company knowledge, propose decisions and coordinate actions. These capabilities are useful when they sit inside a workflow with the right context, permissions and feedback.

An AI feature needs a defined job. What information may it use? Which outputs can be accepted automatically? What requires review? Who handles uncertainty? What happens when a model or an integration is unavailable? These choices turn a promising capability into a usable part of the operation.

We distinguish interpretation from authority. A model can interpret a request while explicit business rules control whether an action is permitted. For consequential actions, the application can require approval, retain an activity record and provide a recovery path. The appropriate design depends on the process and its consequences.

The result should be measured in the work that gets completed: less manual effort, more reliable decisions or better access to information. Adding a conversational interface is only useful when it improves that work.

AI should have a role, a boundary and a measurable contribution.

Reuse the foundations. Design the fit.

Purpose-built software does not require rebuilding every layer.

Custom software is often discussed as if the choice were between buying a complete product and writing everything from scratch. In practice, a tailored application can combine existing cloud services, identity systems, databases, libraries, business platforms and AI models.

The investment belongs in the business-specific layer: workflows, decisions, integration behavior and the experience people use to complete their work. Mature foundations reduce the amount that must be created and maintained. Reusing them is consistent with building a solution that fits one company.

Existing systems can remain the sources of record. A new application can connect them, give a team a coherent workspace and introduce AI capabilities where they are useful. Replacing a core system is a separate decision that should be justified by its own benefits, costs and migration requirements.

This architecture also needs an exit plan. Contracts should clarify access to source code, documentation, data and infrastructure. Integration boundaries and model dependencies should be understandable to the team that will maintain the solution. Tailoring is valuable when it gives the business more control over change.

Shared foundations	Identity, infrastructure, databases, existing systems and models.
Business-specific software	Your workflows, rules, permissions, integrations and experience.
Applied intelligence	Interpretation and assistance inside the work, with appropriate controls.

Reinvent the part that needs to fit. Keep the foundations that already work.

A better build, buy or extend decision.

Choose the smallest commitment that can deliver the required fit.

The decision starts with the operation, not a preferred technology. A standard product may be the best answer for a common need with little differentiation. Extending an existing system may close a specific gap. Building a tailored application may be justified when the workflow is distinctive, the cost of workarounds is material or the business needs control over how it evolves.

Compare alternatives over the same time horizon and scope. Include implementation, configuration, integration, training, ongoing operation, model usage, support, maintenance and expected changes. A low initial development price is not the same as a low cost of ownership.

Fit also has an economic effect. Record the recurring cost of manual work, delays, re-entry and avoidable errors. Estimate conservatively and identify the assumptions. Then test the assumptions in a focused release before committing to a wider program.

A tailored system needs an accountable owner and a credible maintenance path. If those are missing, improving the operating model may be more urgent than commissioning software. AI makes this question more relevant; it does not remove it.

Direction	When it makes sense	What to examine
Buy	The standard workflow already fits.	Configuration, integration, licensing and switching costs.
Extend	A valuable system exists, with a bounded gap.	Available interfaces, data ownership and extension limits.
Build	Business-specific requirements justify dedicated software.	Delivery scope, operational ownership and lifecycle cost.

The best choice earns its place through fit, economics and maintainability.

Measure the work, across the lifecycle.

An application is successful when the operation improves after people start using it.

Agree on the baseline before development. Choose a workflow boundary, a responsible business owner and a small set of measures. Record how long the work takes, how much manual effort it requires, how often it needs correction and whether people can complete it using the current tools.

During delivery, assess working versions with representative tasks and real constraints. Track review effort and rework as well as implementation speed. A faster first draft can still create more work later; a successful release must include integration, validation, deployment and adoption.

After launch, compare the operation with its baseline over a meaningful period. Account for changes in volume, task complexity and staffing. Include human review, support, infrastructure and AI usage in the operating cost. Where the comparison is limited, describe the limitation rather than converting it into a headline claim.

Define in advance what would justify continuing, changing direction or stopping. A useful pilot reduces uncertainty about the business case. Its purpose is to establish whether the approach works in this company, not to confirm a predetermined story about AI.

- Flow: completion time, waiting and the volume of work completed.
- Quality: corrections, exceptions and outcomes accepted by the business.
- Adoption: whether people complete the work through the intended workflow.
- Economics: delivery, review, operation, maintenance and change costs.

A working release is evidence. An improved operation is the outcome.

A new way to deliver software services.

The scarce resource is understanding what is worth building.

Impactia combines business understanding, product design, engineering and applied AI in one service. We use AI to reduce repetitive implementation effort and make iteration more accessible. We embed AI in the resulting application when it helps the operation perform better.

The engagement begins with a defined business priority. We agree on the workflow, required integrations, acceptance criteria and a first delivery scope. The team evaluates working software early, so decisions can be made with something concrete in front of them.

Each stage should produce a useful artifact: a shared understanding of the operation, a working version, a validated release and an agreed plan for support and evolution. Scope, responsibilities, timeline and investment belong in the proposal. Claims about savings or speed must be earned in the project.

Our hypothesis is that this combination can make software that fits available to more companies and more parts of their operations. The service changes from supplying implementation capacity to taking responsibility for translating business intent into a system that works.

Reinventing the wheel is worthwhile when the existing one makes the business take the wrong road. We believe the arrival of AI gives companies a reason to ask the question again, with better tools and a more practical way to find the answer.

Let the business define the software. Use AI to make the fit more feasible.

Sources and scope

This paper presents Impactia's thesis and a proposed decision framework. Published findings are attributed to their sources; they are not Impactia client results. Research reviewed on 21 October 2026.

[1] Cui et al. (2025). Field experiments on AI-assisted software development. Microsoft Research.

[Read the original source](#)

[2] METR (July 2025). Randomized study of experienced open-source developers.

[Read the original source](#)

[3] METR (February 2026). Update on developer-productivity measurement and selection effects.

[Read the original source](#)

[4] DORA (2025). Research on AI-assisted software delivery and organizational capabilities.

[Read the original source](#)

Let's build around your business.

hello@impactia.ai

[impactia.ai](#)