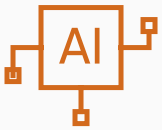
kaizen
SOFTWORKS



AI Adoption Guide

A guide for leaders and teams working through *AI adoption* while the context keeps changing.

It looks at the common mistakes that slow progress and offers a framework for testing ideas, learning from real use cases, and deciding what's worth pursuing.

1 What AI changes

For decades, software was shaped by two constraints: what it could reliably do, and how much effort it took to build. AI is changing both.

Software can now work with ambiguity

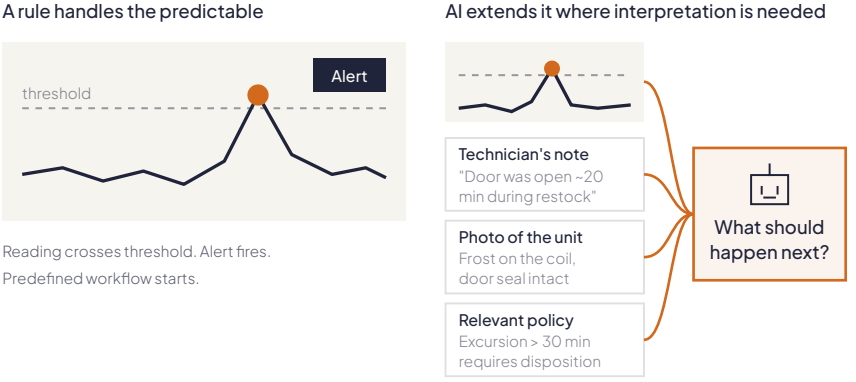
Traditional software automation works best when the inputs, rules, and steps are known in advance.

AI extends software into work that is less structured. It can interpret an email, a document, a photo, a maintenance note, or a question written in plain language.

Take a temperature excursion. Traditional automation can compare a reading against a threshold, trigger an alert, and start a predefined workflow. When the condition is predictable, that is still the right approach.

AI becomes useful when the next step requires interpretation. It can review the temperature history, read a technician's note, interpret an attached photo, check the relevant policy, and help determine what should happen next.

The rule did not become obsolete. The boundary of what software can help with got wider.



A rule and a model doing different parts of the same workflow.

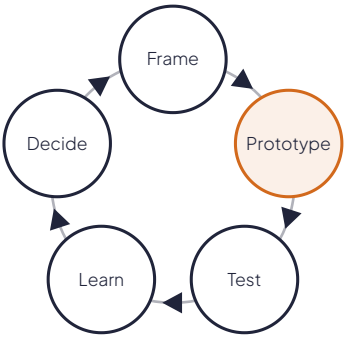
Building is becoming part of finding out

The second shift is in how software gets built. AI is reducing the effort required to design, code, test, debug, and change software. As a result, teams can use working prototypes to answer questions that previously had to be resolved before development started.

Instead of



More work can follow:



When a narrow experiment can be built quickly, you do not need to predict everything correctly before you start. You can build enough to see how the idea behaves, learn from it, and then decide whether it deserves more investment.

Together, these shifts expand what software can do and make more ideas *practical to test.*

2 AI has potential. Value isn't guaranteed

AI will replace everyone. Every company needs agents. Coding is dead. Software is about to be free.

Each statement starts with something real. AI can handle work traditional software struggled with. Agents can execute multi-step tasks. Code can be produced faster. Some software has become much cheaper to create.

The problem is what happens next. Just because something is technically possible does not mean it is reliable enough, economical enough, or useful enough for your operation.

Keep these questions separate

Technically possible ≠ reliable ≠ economically sensible ≠ right for *your business*

Traditional automation may still beat an AI model when the condition is predictable.

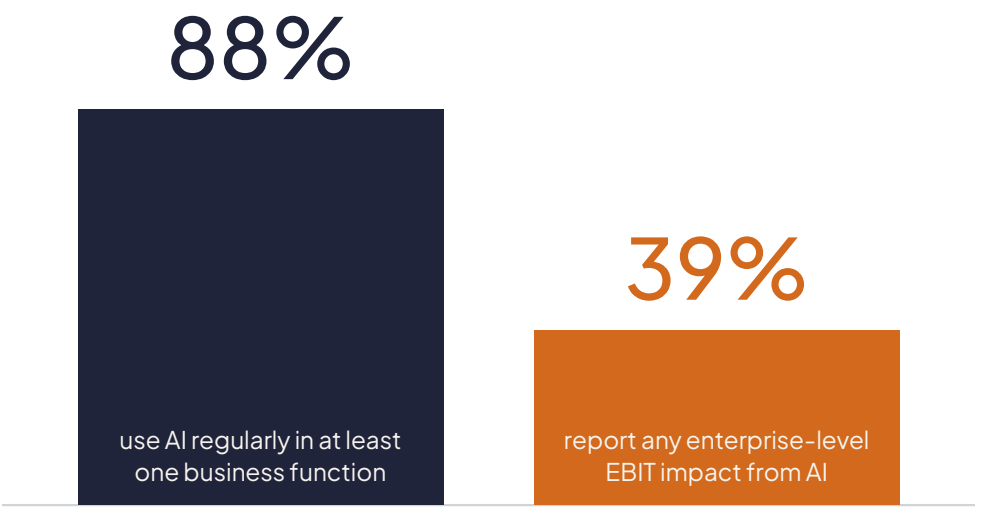
A person may save hours with AI without handing over responsibility for the decision.

An agent may be able to run a workflow, but require so much supervision that the automation is not worth it.

A prototype may look impressive and still fail when it meets real users, real data, edge cases, security requirements, or compliance constraints.

Access has spread faster than impact

Share of organizations, 2025



Source: McKinsey & Company, The State of AI, 2025

McKinsey's 2025 State of AI survey found that 88% of organizations reported regular AI use in at least one business function. Nearly two-thirds had not begun scaling AI across the enterprise, and only 39% reported any enterprise-level EBIT impact from it.[1]

Most organizations have access. Far fewer have found the uses that matter.

So the useful question is not *how much of this will AI replace?* It is: *what has changed enough that we should now do something differently?*

You can answer that one with what you already know: your workflows, your data, your people, your risks and your economics.

3 AI is not a traditional tool

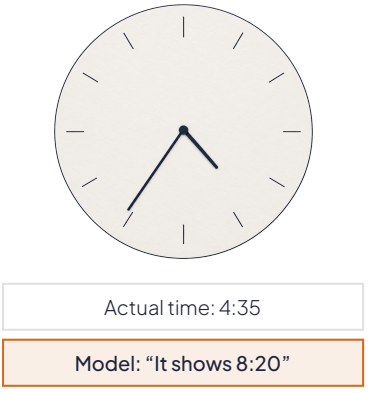
Traditional software taught us to expect predictable behavior: give the system the same input and you should get the same result. AI does not work that way.

Stanford's 2026 AI Index put a memorable number on it. In the same year a frontier model reached gold-medal level at the International Mathematical Olympiad, the best model tested on reading analog clocks got them right about half the time. People manage 90%.[2]

The jagged frontier



Gold-medal level at the International Mathematical Olympiad



Reads an analog clock correctly about half the time. People: 90%.

Same year. Same generation of models. Source: Stanford HAI, AI Index Report 2026

Researchers call this the jagged frontier. Extremely capable in one place, unexpectedly weak a few inches away.

Two things follow from that.

The frontier moves





AI-generated video in 2023 was often easy to recognize and easy to laugh at. A few years later, some generated footage became much harder to distinguish from the real thing.

What AI cannot do today is not necessarily *a permanent limitation*.

Capabilities keep changing. Models improve. Tools improve. Costs change. That makes it risky to treat today's limitations as fixed. Experimentation is one way to keep your understanding current.

The model is only one part of the result

A useful way to think about AI performance is:

RESULT =

MODEL × **CONTEXT** × **TOOLS** × **HUMAN INTERACTION**

The capability you chose | What it has to work with | What it can reach | How you ask, iterate and verify

The three factors you control

The three factors you control

MODEL

The model sets a baseline for what is possible. But its capabilities are uneven and keep changing. A model may be excellent at one task and unreliable at another. Benchmarks and vendor demos can tell you something about general capability. They cannot tell you exactly how the model will behave on your task. You have to test that yourself.

CONTEXT

Context is the information the model has available: instructions, examples, documents, policies, business data, and the situation in front of it. It can completely change the quality of the result. The same model that struggles with a vague question may perform well when it has the relevant policy, the right data, and clear instructions.

TOOLS

Tools determine what AI can reach and what it can do. That might include search, code, APIs, databases, or your internal systems. Tools expand what AI can accomplish. They also increase the consequences of mistakes.

HUMAN INTERACTION

Results also depend on how people use the system. How is the task framed? What context is provided? How do people iterate? What do they verify? Because AI outputs are probabilistic, evaluation matters. Instead of asking only: Does it work? Ask: How often does it work? On which cases? And what do the failures look like?

A better model alone does not guarantee a better result. Context, tools, and human interaction are often just as important, and they are the parts you can shape directly.

This is also why AI rarely comes with a simple manual.

Most software tools have a defined job. AI can be applied across many different kinds of work, and its value often becomes clear only through use.

Someone may try AI once, get a poor result, and conclude that it does not work. But the problem may have been the context, the instructions, or the way the task was framed.

The AI projects we have seen stall rarely stalled because the model could do nothing useful.

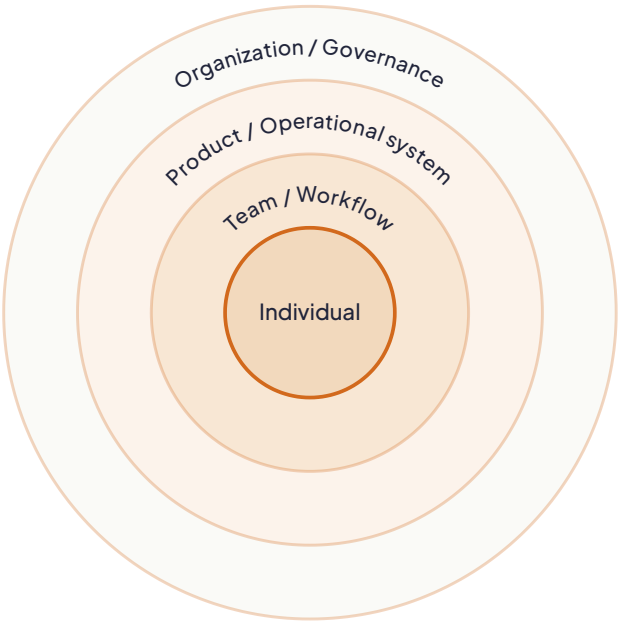
More often, they stalled because the right context was missing, autonomy was introduced before reliability was understood, or nobody had defined how success would be measured.

They rarely stalled *on the model*.

4

AI adoption challenges

AI adoption problems tend to appear at four connected levels. Improving one level while ignoring the others often moves the bottleneck instead of removing it.



Individual
Can people use AI effectively and still know when not to trust it?

Team / Workflow
Is AI changing how the team works, or only making some tasks faster?

Product / Operational system
Can this work reliably in the real world and still create value?

Organization / Governance
Can the organization scale AI without fragmentation and unclear accountability?

A. Individual

Can people use AI effectively and still know when not to trust it?

Overtrust. People accept polished answers too quickly because they sound confident, even when they are wrong. A well-written summary of a corrective action can read perfectly and miss the deviation that mattered.

Undertrust. A few bad results and people stop using AI, including where it would help.

Uneven skills. Results vary widely depending on how people frame tasks, provide context and iterate.

Weak verification. Catching a plausible-looking mistake takes domain knowledge. Not everyone reviewing the output has it.

Cognitive debt. Leaning on AI too much weakens the knowledge and judgment people need to think independently and spot errors.

B. Team / Workflow

Is AI changing how the team works, or only making some tasks faster?

AI without redesign. Teams add AI to an existing workflow and keep the same steps, handoffs and decisions.

Local optimization. One activity gets much faster. The overall process barely changes, because the other steps take as long as before.

Verification bottleneck. AI produces work faster than people can review it. The bottleneck moves from creation to validation.

Unclear handoffs. Nobody has defined what AI does alone, what needs review and when a person takes over.

Uneven adoption. Giving everyone the same tools does not mean everyone gets the same value from them.

C. Product / Operational system

Can this work reliably in the real world and still create value?

Prototype ≠ production. A good demo shows AI can do something. It does not show it will do it reliably with real users, edge cases and scale.

Weak evaluation. Teams rely on "it looks good" instead of defining how to measure quality, errors and acceptable failure rates.

Poor observability. Once live, teams cannot see what the AI is doing, where it fails or whether it is getting worse.

Weak escalation design. It is unclear when AI should stop, ask for help or hand a case to a person.

Hidden operating costs. Inference, human review, rework, monitoring and maintenance can make a technically successful solution too expensive to run.

D. Organization / Governance

Can the organization scale AI without fragmentation, duplication and unclear accountability?

Too many pilots and tools. Teams experiment independently, duplicating effort with little clarity on what to scale or stop.

Unclear ownership. Nobody knows who decides, who is accountable for results or which calls belong to teams versus central governance.

Inconsistent rules. Different standards for models, data, risk and oversight create both gaps and unnecessary friction.

Weak value tracking. Usage and activity are easy to measure. Business impact is not, so it is hard to know what is worth scaling.

Learning stays local. Teams repeat the same experiments because lessons, evaluations and reusable pieces are not shared.

5 Principles for adopting AI

AI adoption is a series of decisions: where AI can create value, what to test, how much autonomy makes sense, and what is worth scaling.

1. Start with a change worth making

Start with a **hypothesis about value**, not with a technology you want to deploy.

What problem are you trying to solve? What outcome should improve? Where are people spending time on work that is repetitive, slow, tedious, or dependent on interpretation?

Then define what should improve: less time, fewer handoffs, better consistency, faster response, or fewer missed exceptions. Then ask whether AI can help.

2. Use prototypes to answer one question at a time

A prototype is valuable because it creates evidence.

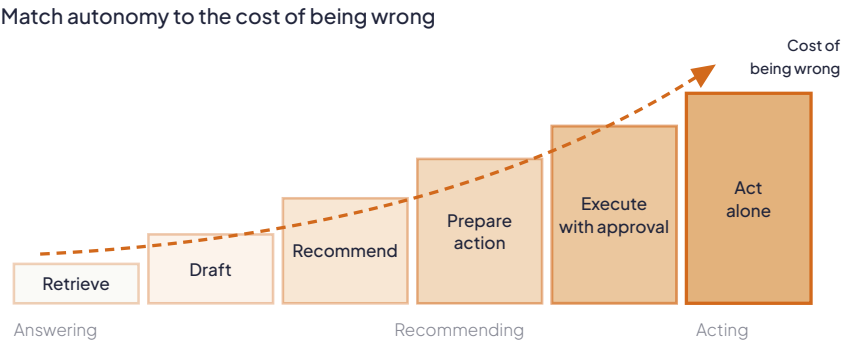
Define what you need to learn, build the smallest experiment that can answer it, and use the result to decide whether to continue, adjust, or stop.

A prototype should answer a question, not quietly become production.

3. Match autonomy to the cost of being wrong

AI can assist, recommend, prepare actions, or act independently.

The higher the consequence of a mistake, the stronger the controls should be: verification, limits, escalation, or human approval. **The goal is not maximum autonomy. It is the right autonomy for the risk.**



Controls should rise with the cost of an error: verification, limits, escalation, human approval, or keeping the decision human.

4. Measure the whole workflow

Making one task faster does not mean the operation improved.

Measure what happens across the process: time, quality, review effort, rework, cost, and the outcome you wanted to change.

The question is not “Is this task faster?” It is “Did the workflow get better?”

5. Learn first. Standardize second

Every experiment should teach you something about the use case and about how your organization adopts AI.

When patterns repeat and value is proven, turn those lessons into shared practices, guardrails, and reusable components.

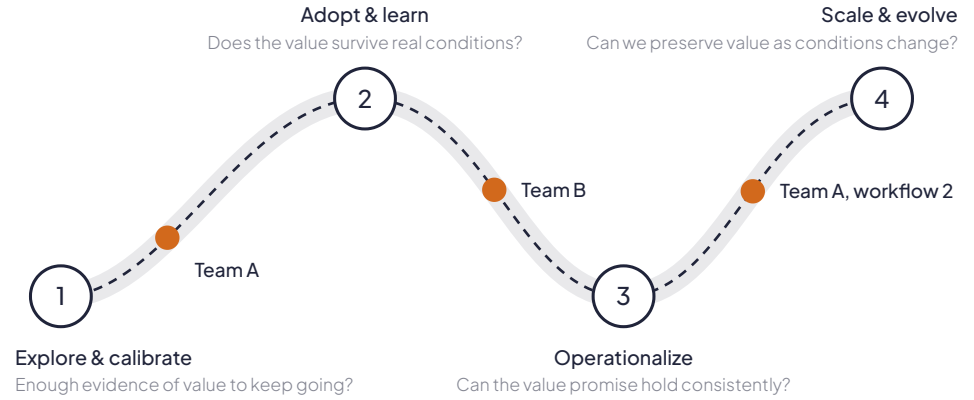
Standardize what the evidence supports.

Takeway: shorten the distance between an assumption and *evidence*.

6 A framwork to adopting AI

This is a lifecycle, not a maturity score. Different teams, and different workflows inside the same team, can sit at different stages at the same time.

An experiment may become a real capability. It may need another iteration. It may expose a problem in the workflow or the data. Or it may show that the idea is simply not worth pursuing. All of those are useful outcomes if they replace assumptions with evidence.



Each marker is one team or workflow, placed at the stage its evidence supports — not at the stage the organization has reached.

Stage 1. Explore and calibrate

The question: Can we find enough evidence of value to keep going?

What it looks like: You decide how much time you are willing to spend reducing the uncertainty, then build the smallest experiment that can test the value hypothesis.

Typical traps: Treating a demo as proof of capability. Looking for a universal recipe. Planning too much before trying anything.

What moves you forward: Enough evidence to believe there is meaningful value worth testing under real conditions. You do not need a complete solution.

Stage 2. Adopt and learn

The question: Does the value survive real conditions?

What it looks like: The capability goes into a real workflow, with real users, real data, normal variation and the exceptions that did not appear in the prototype.

Typical traps: Assuming adoption spreads evenly. Rolling out tools without touching the workflow. Measuring usage instead of value.

What moves you forward: You can describe with evidence where it works, how reliable it is, what human involvement it needs and what value it creates.

Stage 3. Operationalize

The question: Can we make the value promise hold consistently?

What it looks like: A useful field experiment is not yet an operational capability. You build the mechanisms that make the result dependable.

Typical traps: Verification cost eating the savings. Human-in-the-loop as a patch instead of a design. Automating before reliability is known.

What moves you forward: The outcome holds consistently enough to become part of normal operations, with clear ownership and known ways of handling failure.

Stage 4. Scale and evolve

The question: Can we preserve and improve the value as conditions change?

What it looks like: More data, users and connections change cost, latency, reliability and oversight needs. Meanwhile models improve and prices change.

Typical traps: Standardizing too early. Pilot proliferation. Unclear ownership. Treating what was ruled out last year as ruled out for good.

What moves you forward: Repeated learning becomes reusable infrastructure: evaluation methods, guardrails, workflow patterns, policies and clearer ownership.

7 Reflections from our own adoption

We developed this handbook because we needed it for our own transformation.

Our work on AI's capabilities, opportunities, challenges and principles grew out of decisions we were facing across individual practices, team workflows, system design and company-wide governance.

Our focus has been improving the software development lifecycle while exploring how to incorporate AI into the systems we build for clients. Along the way, we have had to adjust both our expectations and how we organize the effort.

We needed enough safeguards for a bounded experiment, not a *complete framework* first.

We had to make room for experimentation

Tool selection, security, governance and access progressed alongside local exploration. People needed dedicated time to investigate and practice. Our Innovation Hub helped organize that work, with pioneers supporting teams and sharing findings.

Waiting for the ideal toolset, a complete governance framework or every training course to be finished would have delayed useful learning.

Short experiments helped us make clearer decisions

Time-boxed efforts gave us checkpoints to inspect results before committing further. Early signals such as output quality, review effort and user feedback helped us assess whether a direction remained promising, even before its full operational impact was measurable.

Experiments also exposed prerequisites: missing information, permissions or resources. Recognizing those gaps helped us distinguish an idea that needed better conditions from one that offered too little value.

A time-boxed experiment buys you a checkpoint, not just a *result*.

Our expectations needed recalibration

Excitement sometimes ran ahead of analysis. When a promising idea delivered less than expected, disappointment could discourage us from investigating why. Discussing results openly helped us examine whether the constraint was the capability, the context, the workflow or the expectation itself.

We also expected to develop a common AI-assisted SDLC approach across clients. The gains proved more variable: project types and team challenges shaped where AI helped.

Improving one task could expose work elsewhere

Automation sometimes moved a bottleneck to another step. In other cases, we made an activity faster before establishing how much it mattered to the overall outcome.

These findings pushed us to revisit handoffs, responsibilities, resource allocation and governance. Progress depended on teams, leadership and supporting functions being able to respond.

Making the work visible helped us learn together

Sharing our direction, activities, findings and observed impact made the effort easier to understand and contribute to. Uncertainty and disappointing results belonged in that conversation too. A stopped experiment could still justify its cost if it resolved an important question.

The handbook reflects where we are today: an approach shaped by *practice*.

Want to run the first one with us?

Sign up for an AI Discovery with Kaizen.

Bring one workflow. We will help you frame what AI could change, identify the data and risks involved, and define what a small experiment would need to prove.

Scan or click here to *book a slot*.

kaizen SOFTWORKS

kzsoftworks.com