

The Return of Owned Software

AI will not kill software. It will change what enterprises rent, what workflows they own, and where the value goes.

ENTRY № 04 · 17 June 2026 · 12 min read

“SaaSocalypse” is the kind of word that is too dramatic to be true and too useful to ignore. The claim behind it is that AI will hollow out software-as-a-service, that the agents will do the work the applications used to do, and that an entire category of recurring revenue is about to evaporate.

The dramatic version is wrong. Software is not dying. The world is still spending heavily on technology, not walking away from it. But the calmer version of the claim is worth taking seriously, because it points at something real.

The more important question is not whether SaaS dies. It is what happens when the cost of creating software falls so sharply that organisations can start owning their workflows again.

The bargain underneath SaaS is being renegotiated. More precisely, the workflow bargain is being renegotiated.

A LEDGER PRINCIPLE

SaaS is not dying. But generic workflows are quietly losing their pricing power.

The bargain we made

For about twenty years, enterprises paid recurring fees because software was hard. Hard to build, hard to host, hard to secure, hard to keep improving. SaaS took all of that complexity and absorbed it. It turned software from a thing you bought into a thing you rented. It turned capital expenditure into operating expenditure. It made upgrades invisible and deployment easy.

That was a good trade, and most of the time it still is.

But the trade came with a quieter cost. In exchange for convenience, enterprises accepted standardisation. They rented increasingly uniform workflows, often priced by the seat, even when the underlying need was narrow, repetitive or highly specific to one organisation. Companies adapted themselves to the software, because adapting the software to themselves was too expensive to contemplate.

The seat was the unit of value because the human was the unit of work. You paid for each person who logged in, because each person was doing the work the software helped them do. As agents take on more of that work, the seat becomes a weaker proxy for value.

Both of those assumptions are now under pressure.

What AI actually changes

The first thing to be clear about is what AI changes and what it does not.

It does not make trust cheap. It does not make compliance free. It does not make a regulator stop caring who signed what. What it changes is narrower and more consequential than the headlines suggest: it lowers the cost of getting from a business need to a working piece of software.

SaaS was built on the scarcity of that path. Useful software required specialised teams, long development cycles, costly infrastructure and deep operational discipline. AI does not remove all of this, but it compresses the distance between intent and a usable system. Agentic coding moves a small internal team from idea to prototype to working tool far faster than before. Authentication, storage, databases, eventing and deployment are reusable primitives rather than things to be built. Language models can read a document, classify it, route it, summarise it and reason over it. And for a great many internal tools, the interface no longer needs to be beautiful. It needs to be fit for purpose, secure and auditable.

A LEDGER PRINCIPLE

Software is becoming less like a finished product and more like a generated response to a specific need.

This is the opening for personal and disposable software. A team needs a tool for one campaign, one reconciliation, one procurement exercise, one internal review. In the old model, that need was too small to justify a product purchase or a formal IT project, so it went unmet or was forced into a generic tool that half-fit. In the AI era, the tool can be generated, used and retired.

That is interesting for individuals. For enterprises, it points at something larger. If building the shape of your own software becomes cheap enough, you start to ask why you were renting that shape from someone else.

The history is not simply reversing

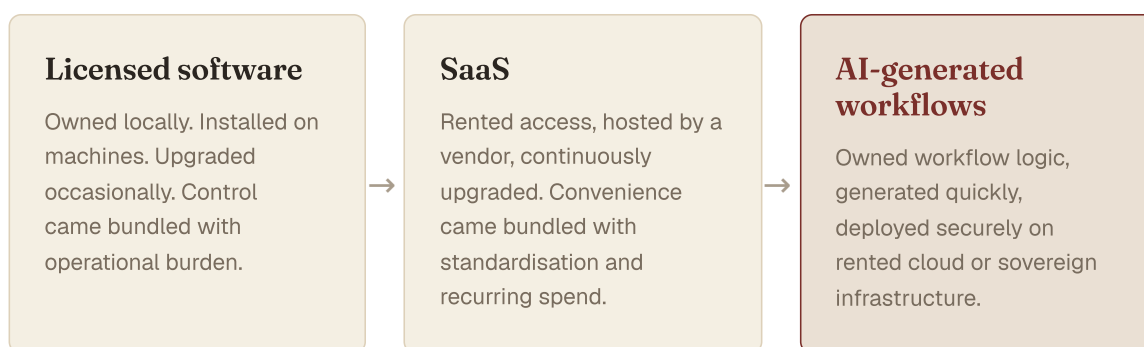
It is tempting to read this as a return to the past. Enterprises rented software, the rented software is now threatened, therefore enterprises will go back to owning software the way they did before.

That is the wrong picture. Nobody is about to buy back the servers, the CDs, the perpetual licences and the monolithic on-premise installs. The operational burden that made SaaS attractive has not gone anywhere.

What returns is not the old object. It is ownership of a particular layer. The return of owned software is really the return of owned workflows.

From Owned Software to Owned Workflows

The history is not simply reversing. Ownership is moving to the workflow layer



Old question: should we buy software or build it?

New question: which workflows should we own?

In the first era, enterprises owned software locally and carried the full weight of running it. In the second, they rented access and let a vendor carry that weight, paying for convenience with standardisation. The third era is not a reversal of the second. It is a recombination: enterprises own the workflow, the business logic, the decision rules and the institutional memory, while renting the infrastructure and the intelligence that let those workflows scale.

The old question was build versus buy. The new question is narrower and sharper. Not whether to build software, but which layer of it is strategic enough to own.

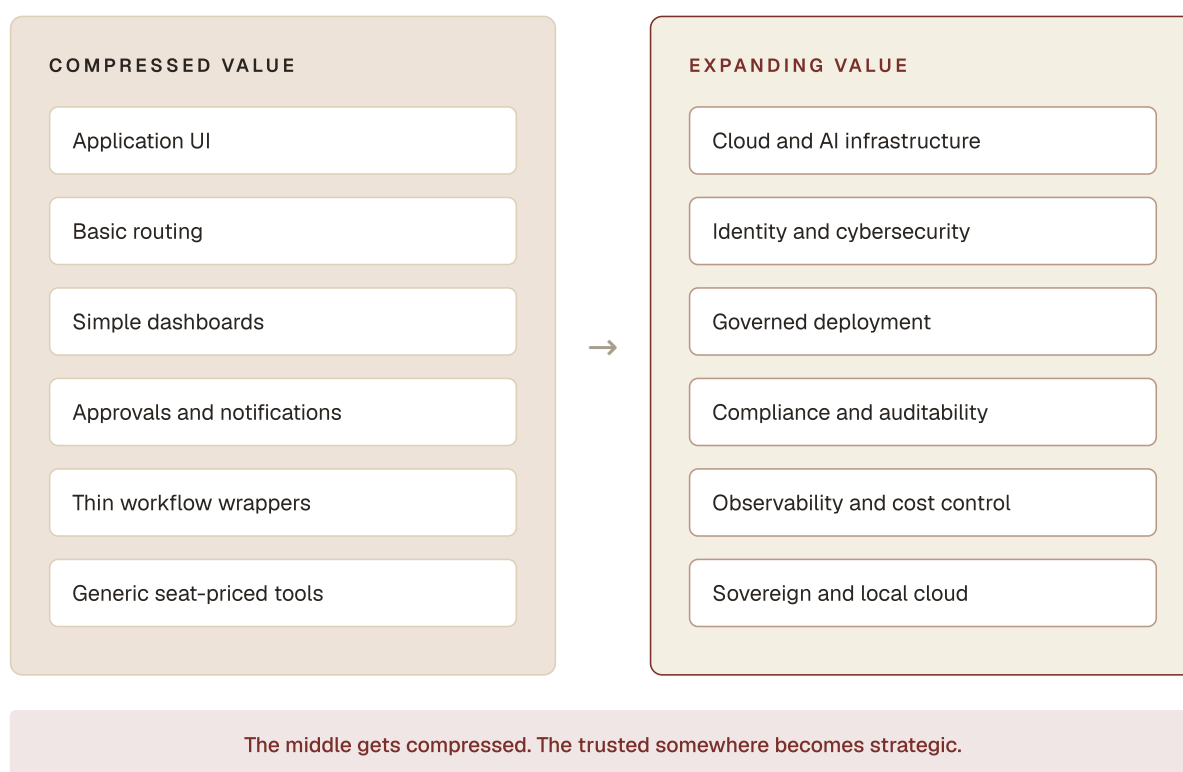
Where the value goes

When generating an application becomes cheap, value does not disappear. It moves.

It moves up, toward outcomes, and down, toward the infrastructure that makes software safe to run. The middle, the generic application layer that mostly captures, routes, approves, notifies and reports, is the part that gets compressed.

When Software Becomes Cheaper, Value Moves

AI compresses the price of generic application work, and expands demand for trusted infrastructure



The market is already pointing this way. Even as application software faces valuation pressure, spending on data centres, cloud and AI infrastructure is rising sharply. Gartner expects worldwide AI spending to grow about 47 percent in 2026, and sovereign cloud spending to grow faster still as AI workflows begin to touch sensitive data. The three largest cloud providers still hold roughly 63 percent of cloud infrastructure spending. The application layer is under pressure at exactly the moment the infrastructure and trust layers are becoming more strategic.

This is the core economic movement. The scarce layer is no longer the ability to generate software. It is the ability to run that software somewhere trusted.

The value of software moves upward into outcomes and downward into infrastructure. The middle gets compressed.

Which software is vulnerable

The mistake at this point is to turn a real signal into a slogan and declare that all SaaS is doomed. The impact of AI is uneven, and the uneven part is where the useful thinking happens.

Two questions separate the software that is exposed from the software that is not.

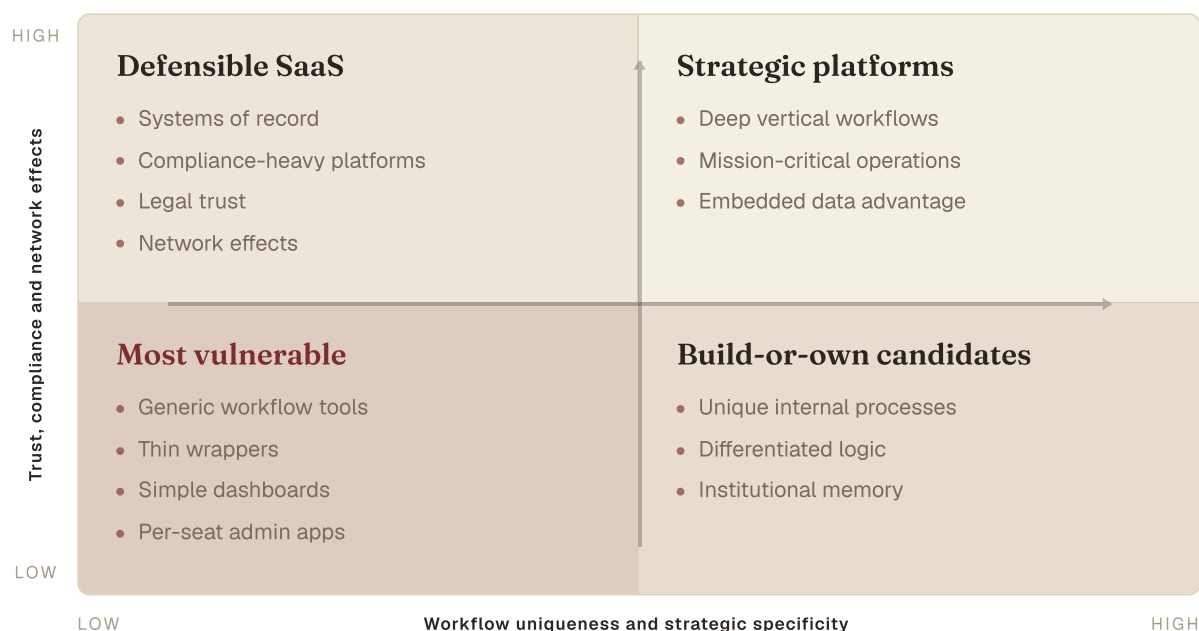
The first is how unique and strategically specific the workflow is. A generic approval flow could belong to any company. A pricing engine that encodes thirty years of one insurer's underwriting judgement could not.

The second is how much trust the software carries. Some software is mostly an interface over common actions. Other software bears legal weight, regulatory certification, network effects or proprietary data that took years to accumulate. That trust is not code, and it cannot be regenerated overnight.

Put those two questions on a pair of axes and the picture stops being a slogan and becomes a map.

Which SaaS Is Vulnerable?

The impact of AI is not evenly distributed across software categories



Vulnerability runs along the diagonal. The more a product depends on trust, proprietary data or a workflow nobody else has, the more defensible it stays.

The most exposed software sits in one corner: low uniqueness, low trust. Generic workflow tools, thin wrappers, simple dashboards, internal admin apps, lightweight trackers. Their value is mostly interface plus routing plus notification, and that is precisely the value an AI-assisted team can now recreate internally. Per-seat pricing made sense when humans were the unit of work. It makes less sense when much of the work is agentic and the workflow itself is cheap to rebuild.

Software high in trust but low in uniqueness remains defensible, but for reasons that have nothing to do with its interface. Systems of record, compliance-heavy platforms, network-effect products. The value is auditability, reliability, liability and ecosystem depth. You do not rebuild that in a weekend, and most organisations are right not to try.

Software high in both is the strongest position of all: deep vertical workflows that are also mission-critical, carrying embedded data and trust at the same time. These are the platforms that are hardest to dislodge.

And then there is the corner that the SaaS era served badly: high uniqueness, lower trust requirement. Workflows that are specific to one enterprise and encode its real institutional advantage, but that do not need a regulator's certification to run. This is the build-or-own corner. For years, organisations bent these processes to fit generic tools because building was too expensive. That constraint is the one AI relaxes.

The biggest threat is not to all SaaS. It is to generic, workflow-light, seat-priced software whose core value an internal team can now recreate.

The nuance, in one example

Digital signatures are the cleanest test of the argument, because at first glance they look obviously vulnerable.

The visible workflow is simple. Upload a document, identify the signers, route it for signature, store the result. Why pay a high per-user subscription for that? An AI-assisted team could rebuild the visible parts in days.

But the visible workflow is not what you are paying for. A serious signature platform also provides identity assurance, legal enforceability, cryptographic integrity, audit trails, trusted timestamping, tamper evidence and the ability to defend a signature in a dispute years later. That is the part that is expensive, and it is expensive for reasons that AI does not touch.

So the category splits cleanly down the middle. The routing, the reminders, the templates and the dashboards drift toward cheap and internally buildable. The identity, the legal defensibility and the compliance stay premium.

This is the pattern in miniature. AI unbundles the SaaS product. Some parts of it become cheap to build, and other parts stay valuable precisely because they carry trust, liability or regulation. The vendors who survive will be the ones who let the cheap parts go and double down on the parts that cannot be regenerated.

Which layer to own

If this is right, the practical question for an enterprise is not whether to build software. It is which layer to own.

Own the Workflow, Rent the Scale

Layer-by-layer ownership, not a full return to on-premise software

Owned by the enterprise

Custom workflows · Internal agents · Business rules · Institutional memory · Process intelligence

Governed by the enterprise

Security policy · Data access · Audit trails · Cost controls · Accountability

Rented or partnered

Foundation models · Cloud services · Databases · APIs · Observability · Developer platforms

Physical and sovereign layer

Data centres · GPUs · Connectivity · Local cloud · Regulated hosting · Power resilience

The enterprise owns what differentiates it, and rents what gives it scale, security and speed.

Own the workflow. The business rules, the decision paths, the internal agents and the institutional memory. This is the layer that defines how the organisation actually operates, and it is the layer worth defending.

Govern the operating environment. Security, identity, data access, audit trails, cost controls and accountability. The enterprise may not write all of this itself, but it must own the governance of it.

Rent or partner for scale. Foundation models, cloud services, databases, developer platforms, observability. For most enterprises, there is little advantage in owning the chip or the foundation model, and a great deal of cost in trying.

Choose the physical layer deliberately. Global cloud, sovereign cloud, local data centres and regulated hosting, depending on what the workflow touches and who is allowed to see it.

The thread running through all four is a single distinction: ownership is not the same as operation. An enterprise can own a workflow without hosting it, and govern an environment it did not build. What matters is holding the layer that encodes its judgement, not running every layer beneath it.

The enterprise owns what differentiates it, and rents what gives it scale, security and speed.

The return of owned software is not a call to rebuild the whole stack. It is a call to be deliberate about which layers are yours.

The new shadow IT

There is a failure mode hiding in this argument, and it is worth naming plainly.

If building software becomes cheap, the risk is not that enterprises build too little. It is that they build too much, with no governance. Every team generates its own tools, on its own assumptions, touching data nobody is tracking, with no owner and no end-of-life. The AI era can produce a new and faster kind of shadow IT, where the clutter is not unsanctioned subscriptions but unsanctioned software.

The answer is not to forbid internal building. That would simply push it underground, which is the worst of both worlds. The answer is to make building governable: a clear intake that defines the problem, the users, the risk level and the expected lifetime of the tool; a deliberate choice between buying, configuring, generating and custom-building; deployment through approved environments with identity, logging and data policy; a named business owner and technical owner for each tool; and a lifecycle that reviews, consolidates and retires.

This is the same lesson I have argued elsewhere about [accountable intelligence](#). When work is produced by a mix of people and software, accountability has to be designed rather than assumed. Generated software is just one more kind of work that someone has to be able to stand behind.

A LEDGER PRINCIPLE

The AI era may create more software than ever. The discipline is preventing it from becoming unmanaged institutional clutter.

The next question: discovery

Governance starts with visibility.

If the AI era makes software easier to create, the next scarce layer is not creation. It is discovery.

An organisation may soon have hundreds of internal tools, generated workflows, agents, copilots and packaged skills. Some will be strategic. Some will be temporary. Some will touch sensitive data. Some will quietly become operationally important. Without a way to name them, find them, assign ownership and understand their risk, ownership without discovery becomes clutter.

This is where the argument continues. If workflows become worth owning again, institutions need a way to make those owned workflows discoverable and accountable. Once software ownership returns, discovery and accountability become unavoidable.

The question worth asking

The strategic shift, reduced to a sentence, is this. The question stops being “which SaaS should we buy?” and becomes “which workflows should we own?”

That second question is harder, because it forces an organisation to know itself. Which of our workflows encode a real advantage, and which are generic plumbing we rent out of habit? Where are we paying per seat when the true unit of value is an outcome or a transaction? Which systems must stay vendor-managed because trust and compliance matter more than control? And where, physically and legally, should our owned workflows be allowed to run?

These are not procurement questions. They are questions about identity. The workflow layer is where institutional knowledge lives, and an organisation that cannot say which of its workflows are strategic does not yet know what it is good at.

Software ownership, redefined

The return of owned software should not be read as nostalgia. It is not a wish to go back to the world of installed applications and perpetual licences. That world had its own burdens, and they were the reason we rented in the first place.

It is something more interesting: a redistribution of value. The generic application gets compressed. The workflow becomes worth owning again. The infrastructure grows more strategic. Governance moves to the centre. And enterprises rediscover an ability

they had quietly given up, the ability to shape software around themselves rather than shaping themselves around software.

The old world asked whether software should be installed or hosted.

The new world asks which parts of software are strategic enough to own, which parts should be rented, and who can be trusted to run the environment in which an organisation's most specific workflows are allowed to operate.

Ownership will matter again, but only if it comes with governance, discovery and accountability.

That is not the death of software.

It is software ownership, redefined for the AI age.

WRITTEN BY

Veemal Gungadin

WITH ASSISTANCE FROM **Claude Opus 4.8** · Anthropic , **GPT-5.5** · OpenAI

Drafted and revised by the author. AI tools were used to sharpen structure and prose; conclusions are the author's own.

Published in *The Goodwill Ledger* at www.goodwill-led.com/essays/return-of-owned-software/