

Who owns an AI system after launch

Ownership of an AI system rarely changes hands on a date. It leaks away over months, until the day something goes wrong and nobody can say whose decision it is.

Author	Amit Govil, Founder and CEO, XONIK
--------	------------------------------------

Published	October 22, 2026
-----------	------------------

Reading time	30 min read
--------------	-------------

Online	https://xonik.com/insights/reports/who-owns-an-ai-system-after-launch/
--------	---

IN BRIEF

An AI system has an owner after launch only when one named person can change it, learns when it is wrong, can switch it off and decides when it is retired, with the time to do each. Otherwise ownership is a name in a register. This report shows how it decays and what holds it in place.

KEY FINDINGS

1. Ownership leaves a system in a fixed order: time, then authority, then information

At launch the project does the job of an owner without anyone calling it that. When the project closes, the name stays in the register while the hours, the mandate and the knowledge go.

2. A name in a register does not make an owner

An owner can show the changes they may approve alone, the view they read and the way they would switch the system off today. Accountability without authority moves blame onto a person and leaves the system as unmanaged as before.

3. Monitoring protects no one until its reader is expected to act on it

A log is kept, a view is looked at, and a read is a view whose reader can start a change. The cycle breaks between reading and deciding, so the first signal is often a complaint from a customer or from a team downstream.

4. A right to stop is empty when the business can no longer run without the system

Once a step is automated the manual route tends to be dismantled, so the owner has nothing to stop into. A fault with no designed route is classed as a defect, a complaint or a one-off, and the question of stopping is never put.

5. A hosted model can change under a live system without telling anyone who decides

The vendor's notice goes to the account, not to a decision maker, and some changes are silent: the service continues and the behavior shifts. Retirement happens on the vendor's date, and the first sign may be a failure.

6. Ownership holds when each hold is written down for one system and tested

One page per system names the owner and a deputy, what they may approve alone, what is read, who can stop it and when it is retired. The owner's time and a running budget have to be agreed beyond the page.

What does this report ask, and why does it matter now?

Once an AI system has passed its launch review, it still needs a person who can change what it does, notice when it is wrong, switch it off and, in the end, retire it. This report asks who that person is in a working organization a year after the project team has left, and what has to be true for the answer to be a person and not a list of functions.

It is written for the executive accountable for systems in production: the chief operating officer, the chief information officer, the head of risk, the head of a business line. They are usually told that every system has an owner, because a register says so. The useful question is whether that owner could act on a Tuesday afternoon when the system is producing wrong answers, and whether they would find out in time.

The question is pressing now. In our delivery experience, systems that began as pilots are now in daily use and their builders have moved on. Hosted models are retired on the vendors' schedule, not the customer's. And public frameworks, and for high-risk systems in the European Union a binding regulation, now describe what an accountable owner holds. The dates from which the high-risk obligations apply moved in 2026, and our briefing on AI regulation sets them out.

A note on terms. By owner we mean the person who holds these for one system: the right to decide changes to it; a regular read on how it is performing, which they are expected to act on; a route for escalation and the right to stop it; and a condition under which it is retired. We use provider and deployer as the EU AI Act defines them. The provider develops the system, or has it developed, and places it on the market or puts it into service under its own name. The deployer uses it under its own authority.

[AI regulation coming into force, and what it changes for enterprise deployments](#)

Method and limits

The report draws on delivery experience and on public documents. The delivery experience is the firm's since 2006, described as pattern and mechanism: what happens to a system's ownership after the project that built it closes. Every engagement is under NDA, so no organization is described and no case is told. The documents were each opened and read on 20 October 2026 for the provisions cited. They are the EU AI Act and the 2026 regulation that amended it, NIST's framework and its generative AI profile, the OECD Recommendation on AI, the UK, Canadian and Singapore documents, and the deprecation documentation of model vendors.

The holds, the review cycle and the ladder of ownership are our working framework, not a published standard.

- It does not measure how common ownership gaps are in any sector or size of organization.
- The findings are patterns seen in delivery, not a sample.

- The EU AI Act's duties for providers and deployers apply to systems classed as high-risk, from dates that moved in 2026. What is said about an article describes its text. How a regulator will read it for a given system is a question for counsel.
- The UK and Canadian documents are written for public bodies, and the NIST, OECD and Singapore documents are voluntary or non-binding. They show what accountable ownership looks like. They are not obligations on a private company.
- Vendor documentation changes, differs between vendors and plans, and may differ from a customer's contract. What is described is the documentation as read on 20 October 2026.
- An owner makes it less likely that a failure goes unnoticed or unanswered. An owner does not prevent failures.

Why does ownership decay after the project team leaves?

Ownership during a project is supplied by the project. There is a sponsor who can say yes, a lead who spends the budget, a steering group that meets, and hours in diaries that exist because the work is funded. At launch these things do the job of an owner without anyone calling it that. The launch review then asks for a named owner, a name is written down, and the project closes.

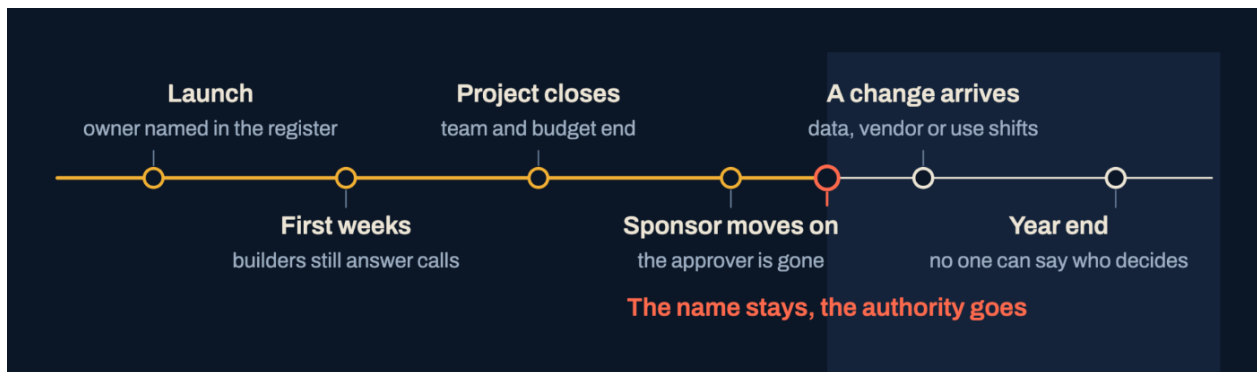
Losses follow, in a fixed order. The first is time. The budget was for building, and running the system has a continuing cost in attention that nobody budgeted: reading reports, answering questions, testing changes. The people who did that work return to other work. The second is authority. The sponsor's mandate carried the project, and the sponsor moves to the next priority or a new role within months. Approvals that used to go to a steering group now have no destination. The third is information. The builders knew where the logs were, why a threshold sat where it did and which cases the model handled badly. That knowledge goes with them to another team and fades.

What remains is the register entry. The person named is usually a business lead who agreed to be listed because the alternative was an empty field. They did not choose the system, did not build it and have no hours set aside for it.

Exhibit 1 follows the first year. Nothing in it is a failure that anyone would log as an event. The name stays on the page while the time, the authority and the information leave, and the first real test arrives with the first change that needs a decision.

EXHIBIT 1

The first year after launch



Nothing in the first year looks like a failure. The authority leaves with the sponsor, and the first change that needs a decision finds a name and nobody able to decide.

How can you tell that ownership has decayed?

These checks show the state of a system without asking anyone's opinion. Is a recurring review of it in anyone's calendar? Is there a budget line for its upkeep, or does every change need a new funding request? Does the contact named in the original plan still hold that role? Are questions about it still sent to a team that has moved to other work?

The cost is that nothing about an AI system stays as it was. NIST's framework observes that these systems may be trained on data that changes over time, sometimes significantly and unexpectedly, and its appendix on how AI risk differs from conventional software adds that they may need more frequent maintenance and corrective triggers because of data, model or concept drift (NIST AI 100-1, overview and Appendix B). An AI system left alone keeps doing what it learned, in a world that has moved.

Singapore's Model [AI Governance](#) Framework (second edition, January 2020, paragraph 3.7) names the mechanism. It asks organizations to transfer knowledge whenever key personnel change, so that staff movement does not open a gap in internal governance, and to review the governance structure when key people change significantly.

Why is a name in a register not an owner?

A register answers one question at one moment: who is accountable? The form is filled in at launch or before an audit, and the easiest name to enter is the senior business person who uses the output or the head of the team that built the system. Neither holds what an owner holds. The head of the building team has the knowledge and not the mandate to change how the business uses the output. The business head has the mandate and no read on performance, no technical route to stop anything and no hours.

Ownership by name and ownership by authority look identical in a register and behave differently in an incident. Exhibit 2 sets them side by side. The right-hand column is not an ideal. Each entry is something a person can be asked to show: the list of changes they may approve alone, the view they read and when, the way they would switch the system off today.

EXHIBIT 2

Owner on paper and owner in practice

What it concerns	Owner on paper	Owner in practice
Who	A name entered at launch	A person who accepted the role, with a named deputy who knows the system
Decisions	Accountable for the system	A written list of changes they may approve alone and of those that go up, and to whom
Time	Added to an existing job	Hours on a fixed rhythm, agreed with their manager as part of the job
Information	Receives a report if someone sends one	Reads a defined view on a fixed rhythm, with a movement that triggers a decision
Stopping	Could ask technology to switch it off	Can switch it off or limit it today, and the business can run without it
The vendor	The contract sits with procurement	Receives the vendor's notices and decides what each one means for the system
Departure	The register is updated at the next audit	A handover to a named successor, and a page reconfirmed on a schedule
Retirement	Not discussed	The conditions are written, and the owner can invoke them

Every row on the right is something an owner can be asked to produce. A row the owner cannot produce is a hold the system does not have.

What does ownership without authority cost?

The difference shows at the moment of a decision. A person who is named but has no authority escalates every decision to a committee, so the system's response to a problem runs at the speed of the committee's calendar. Or they decide anyway, on the business's behalf, without having been given the right, and carry the exposure alone if it goes badly. Accountability without authority moves blame onto a person and leaves the system as unmanaged as before.

The OECD's accountability principle ties accountability to an actor's role and the context, and asks each actor to manage risk according to its ability to act (OECD Recommendation on AI, principle 1.5, as revised in May 2024). The EU AI Act tells a deployer of a high-risk system to entrust human oversight to people who have the skill, the training and the authority for it, and the support to carry it out (Article 26(2)). In both, the capacity to act makes accountability real, not the title.

Accountability without authority moves blame onto a person and leaves the system as unmanaged as before.

Why does monitoring that nobody acts on protect no one?

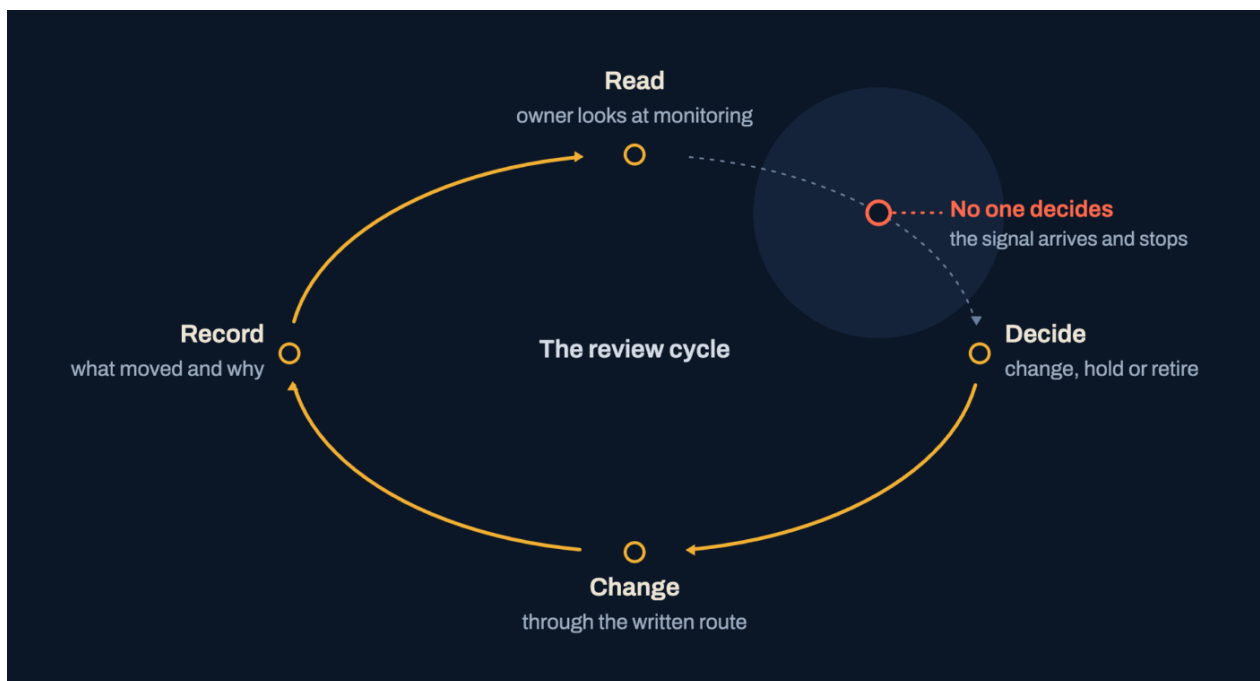
Monitoring is built during the project, as a deliverable, and a deliverable is finished when it exists. Dashboards are live, logs are kept and alerts are configured. After launch the question changes from whether monitoring exists to who reads it, and the answer is often no one in particular. Alerts go to a channel owned by the build team. Reports arrive in a folder. Thresholds were set from the system's first weeks and have not been revisited.

It helps to separate a log, a view and a read. A log is a record that is kept. A view is a record that someone looks at. A read is a view whose reader is expected to act: they know what level of movement matters and they hold the right to start a change. Only a read protects anyone. Monitoring built in a project usually reaches the second stage at best, and it measures what was easy to measure, such as volume, response time and error codes, rather than whether the outputs are still right.

Exhibit 3 shows the review cycle an owner runs: read, decide, change, record. The cycle does not break at the reading. It breaks between reading and deciding. A signal arrives with a person who has no right to settle it, so it is forwarded, discussed and filed. Quality that decays slowly never trips a threshold, so the first signal is often a complaint from a customer or from a team downstream of the system. By then the question is what is wrong and since when, and without a baseline kept from launch nobody can say.

EXHIBIT 3

The review cycle, and where it breaks



The cycle fails at the hand from reading to deciding. Where no one holds the decision, a signal is passed along and the loop never closes.

What do the public documents say about monitoring?

The public documents join the reading to the action. NIST wants the post-deployment monitoring plan to cover how input from users is captured, how a decision is appealed or overridden, and how incidents, recovery, change and decommissioning are handled (MANAGE 4.1). Monitoring and the actions it feeds are one plan. The Canadian Directive on Automated Decision-Making requires scheduled monitoring of outcomes, documentation of feedback, unexpected impacts and human overrides, and the use of those findings to take corrective action (sections 6.3.2 and 6.3.4). The UK AI Playbook asks for a managed release process in which a change can be withdrawn and the system reverted.

The cost of unread monitoring is drift that no one sees, and a system trusted for how long it has run, not for how well it still performs.

Why does the same decision about an AI system come back every quarter?

Whether to widen a live system's use, to accept a known type of error, to change when a person reviews its output, to allow a new data source: each needs someone with the right to settle it and to say what would reopen it. Where that person does not exist, every forum that meets can discuss the question and none can close it.

The general mechanism, why settled decisions return when their reasons and conditions were not recorded, is covered in our perspective on [the decision that comes back every quarter](#). The features specific to AI systems are these. First, the conditions behind a decision change without anyone changing them: a model is updated, the data shifts, the use drifts. A decision to accept an error type was made for a system that no longer quite exists, so reopening it is sometimes right, and nobody holds the test that separates a justified review from a habitual one. Second, the decisions differ in visibility. Approving an expansion is visible and gets a place on an agenda. Deciding not to retrain, or to leave a threshold where it is, leaves no trace and is never put to anyone. Third, an accepted risk is recorded, if at all, without the condition that would end the acceptance.

The cost is a system held in an interim state that nobody chose. Changes queue behind the open question, and because no decision carries a record, a new member of any forum meets it fresh and the cycle restarts.

[The decision that comes back every quarter](#)

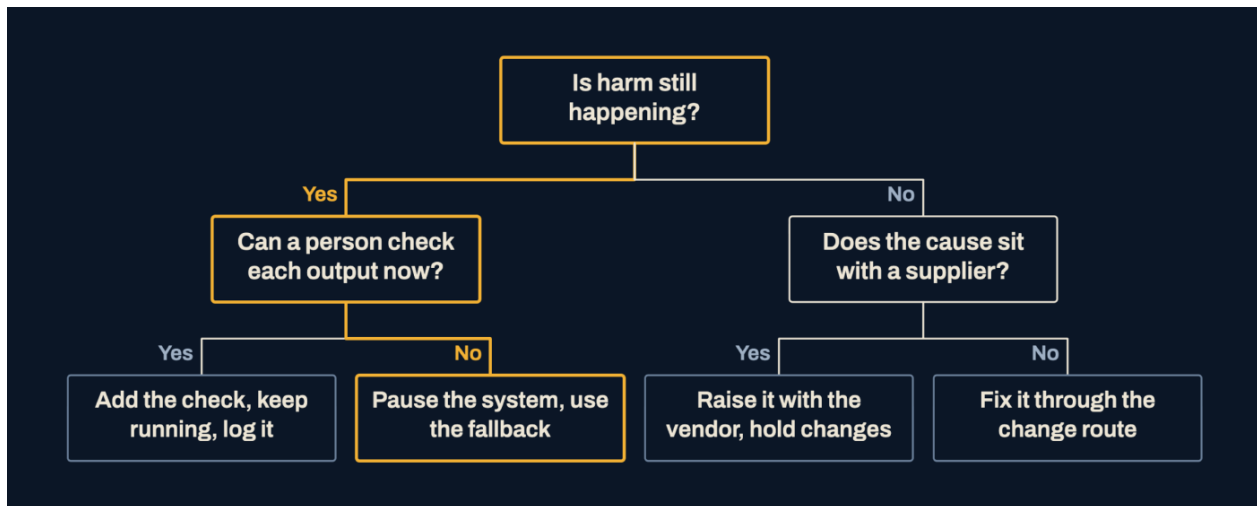
What happens when something goes wrong and no route exists?

The first sign of an AI fault reaches whoever happens to see it: a front-line user, a customer, an analyst who notices that the output looks wrong. They can tell their manager, raise a ticket with the help desk, or contact the team that built the system. None was designed for this. The help desk files a ticket against an application working as designed, because the software ran without error. The builders have moved on. The manager cannot judge severity.

The fault is then classed as a defect, a complaint or a one-off, and each class has a slow path. Severity stays unknown, so nothing is escalated. Escalation needs a person with the right to say stop, and where that person is not named, the question of whether to stop is never put.

Exhibit 4 shows the questions an owner works through when a fault is reported. The first asks whether harm is still happening, which decides between containment and repair. The second asks whether a person can check each output now, which decides whether the system runs on with a check or is paused. Where harm has stopped, the question is whether the cause lies with a supplier.

EXHIBIT 4

What the owner asks when a fault is reported

The tree needs an owner at its root. Each branch is a decision that someone must have the right to take, and the pause branch needs a fallback that works.

Why is the right to stop often empty?

A right to stop is empty unless the business can run without the system. When a system automates a step, the manual route tends to be dismantled: the people who performed it are redeployed, the procedure is not maintained, and the date for switching off the old route has passed. The owner then holds the right to stop but nothing to stop into, and the effective choice is between a faulty system and no service. This is the decision nobody can reverse, not because it is forbidden but because the way back no longer exists.

The UK AI Playbook asks for contingency plans that keep essential services running if a system must be stopped. NIST's generative AI profile asks for tested rollover and fallback, and acknowledges that fallback may include manual processing (NIST AI 600-1, GV-6.2-006). It also asks organizations to define who owns incident response for third-party technologies and to rehearse it (GV-6.2-003).

For a high-risk system under the EU AI Act the route has legal content. A deployer that has reason to consider that use in line with the instructions may present a risk must inform the provider or distributor and the market surveillance authority without undue delay, and must suspend use. A deployer that identifies a serious incident must inform the provider first, then the importer or distributor and the authorities (Article 26(5)). The reporting periods in Article 73 run from the moment the provider or, where applicable, the deployer becomes aware of the incident. For such a system, how quickly news reaches the owner starts a clock.

WHERE IT GOES WRONG

A right to stop with nothing to stop into

The owner holds the right to stop but nothing to stop into, and the effective choice is between a faulty system and no service.

Why do AI systems that should be retired keep running?

Nobody is rewarded for retiring a system. A system that works well enough keeps running because the cost of keeping it is small and spread out, while the cost of retiring it is visible and immediate: someone must replace the function, archive the data, tell the users and change the process around it. The original project budgeted for none of this, and the owner, where there is one, may not hold the right to start it.

What you see is a set of leftovers. Systems whose purpose was quietly changed by their users. Wrapper code nobody has touched, calling a model the vendor has deprecated. Features inside bought software that nobody listed. And systems that nobody dares switch off because nobody can say what depends on them, since the dependencies were never recorded.

Retirement happens anyway, on someone else's date. Anthropic's documentation states that requests to a retired model fail, and OpenAI's that a shut-down model is no longer accessible. An organization that has not decided to retire a system can find the vendor has done it, and the first it hears may be the failure.

The public documents treat the end of life as part of ownership. NIST asks for processes to decommission and phase out systems safely, in a way that does not increase risk (GOVERN 1.7), and its generative AI profile lists what to weigh: data retention, data security after decommissioning, and dependencies upstream and downstream (GV-1.7-002). It also asks organizations to set specific criteria for deactivation and to review them regularly (MG-2.4-004). The OECD's principle on safety says systems should be capable of being overridden, repaired or decommissioned safely (principle 1.4(b)).

The cost is accumulated exposure, ending in a forced exit when the vendor decides.

What happens when a hosted model changes under you?

When a system calls a model that someone else hosts, the deployer owns the system and not the model. The vendor decides when a model is retired, what a model name points to and what changes between versions, and tells the deployer through a channel addressed to an account, not to a decision maker.

The vendors' own documentation, read on 20 October 2026, shows the following. First, notice goes to the account. Anthropic states that it notifies customers with active deployments of upcoming retirements by email. OpenAI states that it notifies customers actively using a model by email, and it exposes an optional shutdown date on each model through its models endpoint, so that a retirement can be watched by a script and not only by reading mail.

Second, a retirement is not the only kind of change. OpenAI's guide to text generation says that even different snapshots of models within one family could produce different results, and recommends pinning production applications to a specific snapshot and building tests that measure behavior, so that performance can be monitored when versions change. Google's Gemini API deprecations page lists some earlier model names whose endpoints are not shutting down and instead automatically route requests to a newer model. A system calling such a name keeps working, and the model behind the name is a different

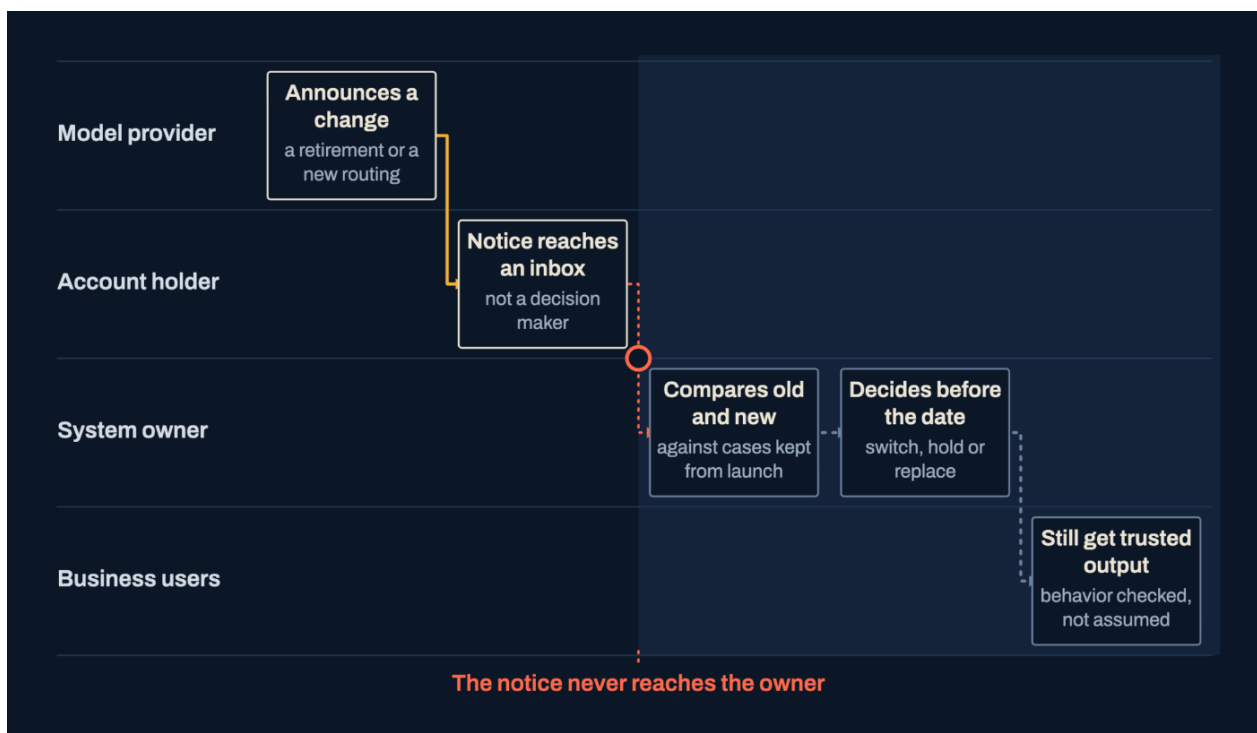
one.

Third, the loud case and the silent case differ. After a retirement date, requests to the retired model fail, and the failure is at least visible. In the silent case the service continues, the behavior shifts and nothing flags it, so when a complaint arrives nobody can say whether the system changed or the world did.

Exhibit 5 follows a change from the provider to the people who rely on the system. The notice lands with whoever opened the account, often a developer or a shared mailbox. Unless a route leads from that address to the owner, the owner learns of the change from its effects.

EXHIBIT 5

A hosted model changes: who sees it, who decides



The owner's steps are sound, but they start only if the notice arrives. The route from the vendor's account to the owner is the part that has to be built.

What does an owner need to keep a hosted model from becoming a hidden decision?

An owner needs a record of which model name and version each system calls, a pinned version wherever the vendor allows it, a notice address that reaches the owner and a deputy, a set of cases with known good answers kept from launch to run against a candidate model, a way back to the earlier setting, and a calendar of every date the vendor has announced.

The Canadian Directive shows what an accountable buyer asks of a supplier. It requires departments to obtain and safeguard all released versions of the software components used in an automated decision system and, where a component is proprietary, to keep the right to access, test and monitor the system, including all released versions of that component (sections 6.2.5 and 6.2.6). NIST's profile for generative AI asks organizations reviewing vendor contracts to consider a clear assignment of liability and responsibility

for incidents and for changes to the system over time, such as fine tuning, drift and decay, (GV-6.2-007).

Where does accountability split between the provider and the deployer?

The EU AI Act divides the work of looking after a high-risk system between the party that supplies it and the party that uses it. The provider develops the system, or has it developed, and supplies it under its own name. The deployer uses it under its own authority (Article 3, points 3 and 4). The obligations of the high-risk chapter follow that line. Exhibit 6 sets out the main ones in plain terms. It is a map for an owner and not a legal analysis.

[Regulation \(EU\) 2024/1689 \(the Artificial Intelligence Act\)](#)

[Regulation \(EU\) 2026/1744 \(Digital Omnibus on AI\)](#)

EXHIBIT 6

What the provider holds and what the deployer holds

Duty	The provider holds	The deployer holds
Oversight by design	Builds oversight measures into the system, or specifies the measures a deployer must put in place (Article 14)	Assigns oversight to people with competence, training, authority and support (Article 26)
Instructions	Writes them: purpose, limits, accuracy, foreseen changes and upkeep (Article 13)	Uses the system as the instructions direct (Article 26)
Watching the system	Runs post-market monitoring over the system's lifetime, using data deployers may supply (Article 72)	Monitors operation against the instructions and informs the provider (Article 26)
Something goes wrong	Corrects, withdraws, disables or recalls; reports serious incidents (Articles 20 and 73)	Suspends use where there is reason to see a risk; informs the provider and authorities (Article 26)
Changes	Manages modifications under its quality system (Article 17); reassesses after a substantial modification (Article 43)	Decides what it changes; a substantial modification can make it the provider (Article 25)
Records	Keeps documentation, and logs under its control (Articles 18 and 19)	Keeps logs under its control (Article 26)
What the Act leaves to the parties	Notice of model changes, support periods and service levels, set by contract	Who inside the organization holds each duty, and when a system is retired

The columns depend on each other. The provider's monitoring draws on what deployers report, and the deployer's oversight depends on what the provider documents.

What does the split mean for an owner?

The division matters to an owner for reasons that show in the detail. The first is that the provider's view of a system depends on the deployer. Article 72 requires the provider to run a post-market monitoring system that actively and systematically collects, documents and analyzes data on the system's performance across its lifetime, and says that data may be provided by deployers. Article 26(5) tells deployers to monitor

operation on the basis of the instructions for use and, where relevant, to inform providers in accordance with Article 72. If a deployer has nobody who reads and reports, the provider's monitoring has a gap where the system meets the world.

The second feature is that responsibility can move. Article 25(1) treats a deployer as the provider of a high-risk system, with the provider's obligations under Article 16, in these cases: it puts its own name or trademark on a system already on the market, subject to the contract; it makes a substantial modification and the system remains high-risk; or it changes the intended purpose of a system that was not high-risk so that it becomes so. The initial provider then ceases to be the provider of that specific system and must cooperate with the new one. In 2026 Regulation (EU) 2026/1744 replaced Article 25(2) to list what that cooperation includes: documentation to assess compliance, information on known limitations and failure modes, and targeted technical access for testing.

Article 3(23) defines a substantial modification as a change, not foreseen in the provider's initial conformity assessment, that affects compliance or changes the intended purpose. Article 13(3)(c) requires the provider's instructions for use to list the changes it pre-determined at that assessment.

For an owner the practical point is that the right to decide a change needs a screen. Before a new use, a fine-tune, a changed workflow or a rebranding goes ahead, someone with the right asks whether it falls inside what the provider foresaw and documented. If it does not, the organization may be about to become the provider, with duties it did not plan for, and nobody decided that. The screen is only as good as the owner's access to the provider's documentation, which is one more reason the vendor relationship belongs to the owner.

General-purpose models add a layer. Article 53(1)(b) requires providers of general-purpose AI models to keep documentation up to date for the providers of AI systems that integrate the model. Article 25(4), as amended in 2026, makes the provider of a high-risk system and any third party supplying a model, tools or components for it settle in writing what information, access and help the provider will get. What a customer receives when a hosted model changes depends on its position under the Act and on its contract. How these provisions apply to a given system is a question for counsel.

What do the public frameworks expect of an owner?

The documents carry different force. NIST's framework and Singapore's are voluntary. The OECD Recommendation is an intergovernmental recommendation, not a law. The UK AI Playbook is guidance for public bodies, Canada's Directive binds the federal institutions it covers, and the EU AI Act binds providers and deployers of the systems it classes as high-risk. Read for what an owner holds, they converge on a common set of expectations.

Authority is named. Canada's Directive makes the assistant deputy minister responsible for the business area that uses the system, or another senior official the deputy head names, answerable for the duties it lists (section 6). The UK Playbook asks for a named senior responsible owner for AI used in a project, and for teams to establish who has the authority to change the model's code. NIST asks that roles and lines of communication for managing AI risk be documented and clear (GOVERN 2.1) and that executive leadership take responsibility for decisions about AI risk (GOVERN 2.3).

Monitoring reaches someone who can act. Singapore's framework asks that the appropriate level of management be made aware of how deployed AI is performing (paragraph 3.7), and NIST that periodic

review be planned with roles defined and its frequency set (GOVERN 1.5).

Change is controlled. Canada requires the algorithmic impact assessment to be updated on a schedule and when the system's functionality or scope changes (section 6.1.3). The UK Playbook requires testing and validation of any update as part of change control.

Stopping and ending are assigned. NIST wants working mechanisms, and named responsibilities, for replacing, disconnecting or deactivating a system whose results depart from its intended use (MANAGE 2.4), and for decommissioning processes (GOVERN 1.7).

Handover is planned. The UK Playbook asks teams to set out how the model will be maintained over time and to plan knowledge transfer and training of new and existing staff, which echoes Singapore's paragraph 3.7.

What the documents leave out matters as much. None says who by title, the hours the role takes, or how a system moves to the next owner when a person leaves. They set the standard and leave the staffing to the organization, which is where ownership fails.

[NIST AI 100-1: AI Risk Management Framework 1.0](#)

[OECD Recommendation of the Council on Artificial Intelligence](#)

[Directive on Automated Decision-Making, Government of Canada](#)

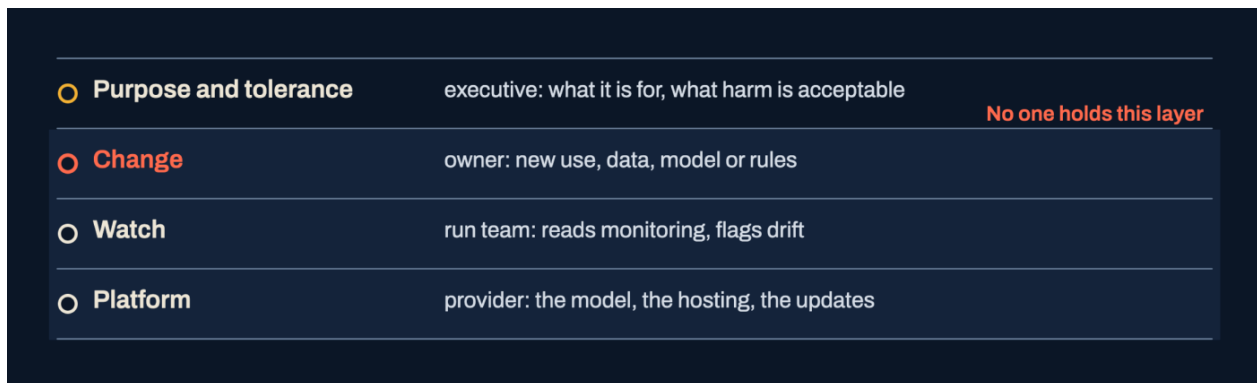
[Model Artificial Intelligence Governance Framework, Second Edition, IMDA and PDPC Singapore, 21 January 2020: paragraph 3.7](#)

What changes the outcome?

The outcome changes when ownership stops being a label and becomes a set of holds that can be tested one by one. For each live system the organization writes down who holds each of them, and checks that the person can in fact use them. Exhibit 7 shows the layers of decision rights this involves. Executives decide what the system is for and how much harm is tolerable. The run team watches. The provider supplies the platform and its updates. The layer between them, the right to change the system's behavior, use, data and model, is in our delivery experience the one that goes missing after launch. Without a holder, the watching layer produces information with nowhere to go, and the platform layer changes on someone else's terms.

EXHIBIT 7

Layers of decision rights over a live system



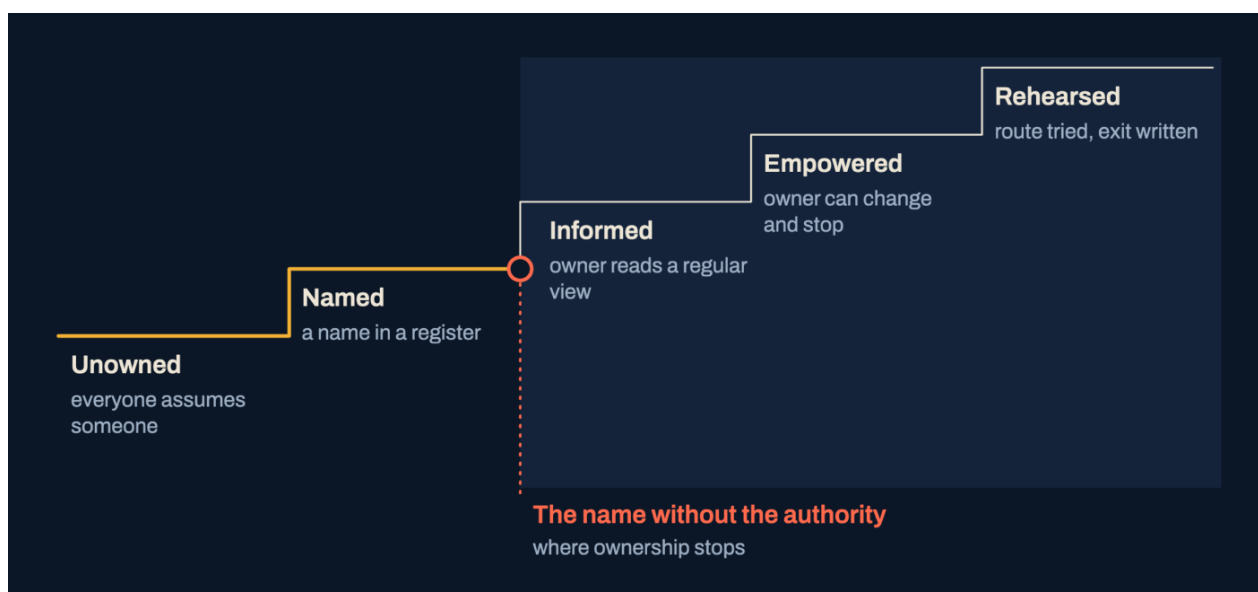
Every layer except one has a natural holder. The change layer is the one that has to be assigned on purpose, and without it the layers beneath it have no one to report to.

How do you tell how far a system's ownership has come?

Ownership is a matter of degree, and a system can be placed on a ladder. Exhibit 8 sets out the levels. At the lowest the system is unowned and everyone assumes someone else holds it. Naming a person is the first step up, and it is the step at which a register can be marked complete. The owner is then informed, receiving a regular view. Next the owner is empowered, with the right to change the system and to stop it. At the top ownership is rehearsed: the escalation route has been tried, a successor is named, and the conditions for retirement are written. The levels are our working model, not a published standard, and a system sits at the level of its weakest hold.

EXHIBIT 8

A ladder of ownership



A register can show a system as owned at the second step. The steps that make ownership useful are the ones above it.

What goes on the ownership page, and what must be true beyond it?

The working document is one page per system, kept where changes are made. It names the owner and a deputy. It lists the changes the owner may approve alone and those that go up, and to whom. It says what is read, how often, and what movement triggers a decision. It states who can stop the system, how, and what the business does meanwhile. It records the model version the system calls and where the vendor's notices go. It states when the system is retired, who takes over if the owner leaves, and the date the page was last confirmed.

Beyond the page, these things decide whether it works. The owner's time must be agreed by the owner's manager, as part of an actual job. The cost of running the system, which means monitoring, review, vendor changes and the occasional rebuild, needs a budget that survives the project, because it cannot be demanded after launch from a team that was never funded to supply it. And the sponsor's authority must pass to someone: when a sponsor leaves, an executive takes the top layer.

What changes in the business is modest and observable. A fault reaches a person the day it is seen. A vendor notice is read as a decision. Decisions are recorded with the condition that would reopen them, so they stay decided. A departure triggers a handover and not a search. Retirement happens on the owner's schedule and not the vendor's. None of this prevents a system from failing. It reduces the chance that a failure goes unnoticed or unanswered.

What is the step-by-step response?

1. List the systems that are live

The head of risk or the office of the chief information officer builds a list with each business function, covering systems built in house, systems bought and AI features inside bought products. Each entry records the model and version called, the vendor, the users and the decisions touched. At the end there is one list.

2. Order them by consequence

The risk lead and the business heads rank the systems by what happens when the output is wrong and who is affected. Systems whose wrong output reaches a person outside the team come first. At the end there is an order of work.

3. Test each hold in a room

For one system at a time, the proposed owner, their manager, a technology lead and whoever manages the vendor sit together. The owner shows the changes they can approve alone, the view they read, how they would switch the system off today and when it is retired. Each gap is the work. At the end there is a one-page ownership record, agreed by the owner and by their manager, who also agrees the time.

4. Turn monitoring into a read

The owner and the run team choose what is measured, who looks, how often and what movement triggers a decision, and they keep a baseline from launch. At the end there is a view with thresholds, a

baseline and a written rule for what happens when one is crossed.

5. Draw the route and rehearse the stop

The owner, operations and legal draw the route from the first sign of a fault to the person who can stop the system, and settle what the business does meanwhile. They try it once on a rehearsed case. At the end there is a route card, a working fallback and a record of the rehearsal.

6. Bring the vendor to the owner

The owner and procurement point the vendor's notice address at a role mailbox that the owner and a deputy read, pin the model version where the vendor allows it, assemble cases with known good answers and calendar the vendor's dates. They ask for contract terms on notice and version support. At the end the notice route is tested.

7. Write the exit and the handover

The owner and the sponsor write the conditions for retirement and the screen for changes that may fall outside what the provider foresaw. They name a successor and set a date to reconfirm the page. At the end the page says when the system ends and who takes over.

8. Run the cycle and review the roles

The executive sponsor sets the review rhythm and reads, at the same rhythm, where each system stands on the ladder. Departures and reorganizations trigger a review of the pages they touch. At the end there is a record of decisions and a ladder level for each system.

How can you check your own organization?

- For each live AI system, including AI built into bought products, I can name one person who can change it without asking a committee, and a deputy.
- That person has accepted the role, and their manager has agreed the time as part of their job.
- The owner reads a defined view of performance on a fixed rhythm, against a baseline from launch, and the movement that triggers a decision is written down.
- Someone can switch the system off or limit it today, and the business can run without it.
- The route from the first sign of a fault to the owner is written down and has been tried.
- Vendor notices about the model reach the owner, and not only the person who opened the account.
- We know which model name and version each system calls, and whether the version is pinned.
- A change of use, data source or model version passes a screen that asks whether it falls inside what the provider foresaw.
- Each decision about the system is recorded with its reason and the condition that would reopen it.
- The conditions under which the system is retired are written, and the owner can invoke them.
- When the last owner or sponsor left, a handover happened and the page was reconfirmed.
- The cost of running the system has a budget line that continues after the project closes.

What do readers ask about owning a live AI system?

What makes someone an owner of a live AI system and not just a name in a register?

An owner can show the changes they may approve alone, the view of performance they read and act on, how they would switch the system off today, and the condition under which it is retired, with time set aside for each. A name in a register supplies none of these.

Why does ownership fade after the project team leaves?

The project supplied the owner's time, authority and information without anyone naming it. When the project closes, the hours go first, then the mandate as the sponsor moves on, then the knowledge as the builders leave. The name stays in the register, and nothing is logged as an event.

Who is responsible when a hosted model changes or is retired?

The deployer owns the system, not the model. The vendor decides when a model changes or retires, and its notice goes to the account, not a decision maker. The owner needs a notice address that reaches them and a deputy, and a record of the version called.

Does a law require every AI system to have a named owner?

It depends on the document and the system. The frameworks differ in force: some are voluntary, some are guidance for public bodies, and the EU AI Act binds providers and deployers only of systems it classes as high-risk. Whether a rule binds yours is a question for counsel.

Sources and notes

All sources were read on 20 October 2026 for the provisions cited in the text. The EU texts were read in the English version published by the Publications Office of the European Union.

1. [Regulation \(EU\) 2024/1689 \(Artificial Intelligence Act\), Official Journal of 12 July 2024: Articles 3, 9, 13, 14, 16 to 20, 25, 26, 43, 53, 72 and 73](#)
2. [Regulation \(EU\) 2026/1744 of 8 July 2026 \(Digital Omnibus on AI\), amending Articles 4, 25, 72 and 113 of Regulation \(EU\) 2024/1689](#)
3. [The AI RMF 1.0 \(NIST AI 100-1\), NIST, January 2023: overview, GOVERN, MANAGE and Appendix B](#)
4. [NIST AI 600-1: Artificial Intelligence Risk Management Framework, Generative Artificial Intelligence Profile, July 2024](#)
5. [OECD Recommendation of the Council on Artificial Intelligence, OECD/LEGAL/0449, as revised on 3 May 2024: principles 1.4 and 1.5](#)
6. [Artificial Intelligence Playbook for the UK Government, GOV.UK, published 10 February 2025: accountability, governance structures and operational monitoring](#)
7. [Directive on Automated Decision-Making, Treasury Board of Canada Secretariat, last modified 24 June 2025: sections 6.1 to 6.3](#)
8. [Model Artificial Intelligence Governance Framework, Second Edition, IMDA and PDPC Singapore, 21 January 2020: paragraph 3.7](#)

9. [Model deprecations, Claude Platform Docs, Anthropic](#)
10. [Deprecations, OpenAI API documentation](#)
11. [Text generation, OpenAI API documentation: prompt engineering and model snapshots](#)
12. [Gemini deprecations, Gemini API, Google AI for Developers \(page last updated 20 October 2026\)](#)

USE THIS

[Find the gaps with the AI governance assessment](#)

[Start from the AI governance policy template](#)

Choose one live AI system and ask its owner to show each hold in one room: the changes they can approve alone, the view they read, how they would switch it off today, and the condition under which it is retired. The first one they cannot show is where to start.

This document was made from the article at <https://xonik.com/insights/reports/who-owns-an-ai-system-after-launch/>. The online article is the current version.