



Satya's practice focuses on technology transactions, intellectual property, and commercial agreements for startup and growth-stage technology companies across AI, robotics, cloud computing, and emerging tech sectors.

*Email: snarayan@gcalaw.com
LinkedIn Newsletter: Techline Brief
Connect on LinkedIn*

Business Continuity in Agentic AI: Escrow Is a Control Problem

Satya S. Narayan

October 20, 2026

Most customers negotiating AI continuity rights begin by asking for model-weight escrow. That is understandable. Weights are the closest analogue to source code, and source code escrow has been negotiated for decades. In a well-drafted software escrow arrangement, the customer typically receives the released code under a limited, time-bound license that permits support, maintenance, bug fixes, and workarounds, but not the development of a competing product.

But model-weight escrow assumes that continuity depends primarily on controlling access to the model itself. In modern agentic systems, that assumption is often wrong.

The continuity challenge in AI systems is rarely just a model problem. It is a control problem.

The framework below is written from the licensor's perspective. It seeks to preserve the value of the technology while still providing customers with the continuity they are seeking: a service that keeps running long enough to finish the contract term or transition to another solution.

For sophisticated AI deployments, particularly agentic systems, there are at least three distinct assets:

- The model itself (weights or model package)
- The harness (orchestration logic, prompts, retrieval systems, routing rules, memory architecture, workflows, tool integrations, and related software)
- The operational capability required to deploy, monitor, maintain, and evolve the system

These assets present different risks if released and provide different continuity value to the customer. Treating them as interchangeable is a mistake.

Why Traditional Software Escrow Doesn't Fit AI

Software escrow works because several assumptions generally hold.

There is usually a meaningful distinction between maintenance (e.g., bug fixes, security patches, compatibility updates) and enhancement. Modified code can be compared against the deposited version, making derivative works easier to identify. Even as increasingly capable AI-assisted development tools reduce some of the practical friction associated with working from released source code, software modifications generally remain easier to trace than many forms of model adaptation or harness modification. Customers also typically cannot transform escrowed source code into a fundamentally different product overnight.

Those assumptions do not translate neatly to AI.

Model weights offer far less practical friction. Once weights are available, a sophisticated recipient may be able to fine-tune, distill, or otherwise build upon them much faster than a recipient working from source code alone.

Fine-tuning modifies the original weights, creating a directly derived model whose lineage may be detectable. Distillation is different. A new "student" model is trained to imitate the outputs of a "teacher" model, often using an entirely different architecture. Nothing is copied directly. The result strains derivative-works concepts that evolved around software code, and it remains unsettled whether model weights generated through automated training processes attract meaningful copyright protection. For licensors, contract and trade-secret protections often become the primary safeguards.

Patent-blocking concerns also look different in AI. Courts and patent offices remain skeptical of claims that merely apply known machine-learning techniques to new domains, though system-level innovations may be patentable. The risk is real, but usually longer-term.

Those concerns matter. They are not, however, the hardest escrow problem in agentic AI.

The Bigger Issue Is the Harness

Customers often focus on the model because it resembles source code. In practice, the harness is frequently more valuable.

The harness includes orchestration, prompts, retrieval architecture, routing logic, memory systems, workflow design, tool integrations, and evaluation frameworks. It determines how the model behaves in production.

Consider an insurance claims agent built on a general-purpose model. The model provides the underlying reasoning capability. The harness is the platform layer: the retrieval architecture, orchestration logic, routing framework, workflow engine, and evaluation systems developed by the licensor. When configured for a particular insurer, the harness enables the model to access the insurer's policy repositories, fraud-detection systems, and claims-management tools through APIs or other controlled interfaces, while those systems and the underlying data remain under the insurer's control. The model determines how to analyze a claim or respond to a user request; the harness determines what information the model can access, what tools it can use, and what actions it can take. Replace the model with a comparable one and the deployed solution may survive with some retuning. Remove the harness and what remains is a general-purpose model disconnected from the operational architecture that makes the solution useful.

Ownership also complicates the analysis.

Many licensors deploy agentic AI solutions using foundation models owned and operated by third-party model providers. A licensor using a proprietary foundation model through an API may never possess the weights, even if it fine-tunes the model on the model provider's platform. In those cases, there may be little or nothing available to escrow on the model side beyond configuration data and training artifacts. Continuity depends on upstream agreements that may not survive the licensor's insolvency and may, depending on their terms, require the consent of the underlying model provider before they can be assigned.

By contrast, a licensor building on open-weight models can deposit weights, but any release remains subject to the underlying license terms.

In either scenario, the harness may be the most significant AI asset the licensor actually owns.

When Maintenance Starts Looking Like Innovation

Traditional escrow arrangements depend heavily on the distinction between maintaining a system and creating new functionality.

In agentic systems, that distinction quickly becomes blurry.

- Adding a new tool connector can enable entirely new actions.
- Updating routing logic can unlock new workflows.
- Expanding retrieval access can materially improve capabilities.
- Modifying memory architecture can enable personalization and more sophisticated interactions.
- Revising prompts can significantly change reasoning, tool use, and workflow execution.

A customer may reasonably characterize those changes as maintenance while substantially expanding what the system can do.

As a result, a maintenance-only license can appear narrow on paper but provide significant opportunities for capability expansion in practice.

These changes are also difficult to police. The evidentiary trail is less clear than in traditional software development, and the line between maintenance and enhancement is harder to identify.

The Goal Should Be Continuity, Not Technology Transfer

Many escrow discussions begin with a straightforward question:

What should be released if the licensor goes bankrupt?

For agentic AI, the better question is:

What level of control is actually required to ensure business continuity?

In many cases, the answer is surprisingly little.

Most customers do not need access to the underlying technology. They need the service to keep operating.

Bankruptcy law only partially addresses this problem. Section 365(n) of the U.S. Bankruptcy Code allows a licensee to elect to retain rights to licensed intellectual property if a debtor rejects the license. It does not require continued hosting, support, maintenance, or updates. A customer may retain legal rights to technology it cannot realistically operate.

The challenge is preserving continuity without unnecessarily transferring the licensor's most valuable intellectual property.

Continuity Is a Bridge, Not a Perpetual License

Continuity rights should facilitate transition, not create permanent access.

A smart licensor does not grant perpetual rights in released escrow materials. The release license should expire at the earlier of a replacement migration, the end of the contract term, or a defined outside date. Upon expiration, escrow materials should be returned or destroyed, use rights should cease, and continuity-related rights should terminate.

The customer receives a predictable runway. Future owners of the technology receive certainty that continuity rights will not become a long-term encumbrance.

A Better Framework: Continuity Through a Successor Operator

The evolution mirrors what happened in SaaS.

Early SaaS customers often demanded source code escrow because they viewed cloud services through a traditional software-licensing lens. Over time, many recognized that source code alone did not deliver meaningful continuity. Modern platforms require infrastructure, deployment expertise, operational procedures, and specialized personnel.

Agentic AI may represent the next stage of that evolution.

A successor operator may be a managed services provider or other pre-approved technical operator designated before a release event. The operator's role is to preserve continuity without transferring ownership or control of the underlying technology to the customer.

Tier 1: Endpoint Continuity

The simplest continuity model preserves inference services.

The deployment remains running in a transferable environment, such as a dedicated cloud account or a customer-controlled environment where the licensor's assets remain behind a managed endpoint.

Following a release trigger, an escrow provider or designated successor operator assumes control and continues operating the deployment.

The customer receives continuity of service.

The customer does not receive:

- Weights
- Source code
- Harness access
- Model-development rights

For many deployments, that may be sufficient, but only for a short runway: certificates expire, runtimes are retired, and upstream model APIs are deprecated. Who pays the cloud bill, and who holds credentials and encryption keys after a trigger, should be settled at signing.

Tier 2: Managed Continuity

Where the system requires maintenance, security updates, or redeployment, the parties can designate an independent successor operator to receive the materials from the escrow agent as necessary to maintain and operate the platform:

- Weights
- Harness materials
- Deployment artifacts
- Operational documentation
- Runbooks

The customer still receives a service rather than direct access to the underlying assets.

Because the successor operator receives highly sensitive materials, licensors should pre-approve operators, exclude competitors and their affiliates, and impose strict confidentiality and use restrictions.

Tier 3: Customer-Hosted Deployment

Some customers have legitimate requirements for local deployment due to regulatory, security, or data-sovereignty concerns.

Even then, direct release of source code and unencrypted model weights should not be the default.

Instead, the successor operator can provide:

- Compiled software packages
- Model deployments
- Encrypted weights
- Updates and fixes
- Replacement model packages

The customer controls the infrastructure and data while the technology remains protected.

Encryption, however, is not a complete solution. Models must eventually be decrypted in memory to run, and prompts necessarily reach the model in usable form. Confidential computing and secure enclaves can reduce risk, but they cannot eliminate it.

Tier 3 also creates the most favorable environment for model distillation.

There is no meaningful fourth tier. At some point, direct access to source code, harness materials, and unencrypted weights ceases to be continuity and becomes acquisition of the technology itself.

Distillation: The Leak That Doesn't Need the Weights

Most escrow protections focus on controlling access to assets.

Distillation sidesteps that premise.

Anyone with sufficient access to system outputs may attempt to replicate aspects of the solution without ever receiving weights or source code.

The risk extends beyond the model itself. Agentic systems generate traces, retrieval patterns, tool calls, routing decisions, and workflow behavior. At sufficient scale, those signals can teach another system to imitate the harness, which may be the more valuable asset.

In Tiers 1 and 2, operators can mitigate risk through rate limiting, monitoring, and withholding information that makes distillation more efficient.

Ultimately, however, the strongest protections remain contractual:

- Prohibitions on training, fine-tuning, or distilling competing systems from outputs or traces
- Usage limits tied to historical activity
- Periodic compliance certifications
- Acknowledgment that violations may cause irreparable harm and support injunctive relief

Customers should be free to use outputs to run their businesses, but not to train replacements for the licensed solution.

Protecting Improvements and Preventing Value Leakage

Patent blocking remains a real longer-term risk. During the continuity period itself, three nearer-term risks are more likely to materialize:

- ***Exceeding the license scope.*** The customer ends up with a more capable system than it licensed or paid for. That is a contract problem whether or not anything is patentable.
- ***Trade secret leakage and competitive learning.*** The customer or the successor operator learns how the harness works and builds independently, and the licensor or its acquirer will struggle to prove otherwise.
- ***Clean title.*** In a bankruptcy sale under Section 363 of the U.S. Bankruptcy Code, buyers pay for clean intellectual property. If ownership of continuity-period improvements is ambiguous, potential acquirers may discount their valuation of the asset to account for litigation risk, ownership disputes, or limits on future exploitation.

A continuity framework should clearly address ownership of:

- Fine-tuning outputs
- Workflow enhancements
- Prompt libraries
- Retrieval improvements
- Tool integrations
- Patentable inventions
- Other developments derived from escrowed technology

Customers and operators will inevitably retain some general know-how. Rather than attempting to eliminate all residual knowledge, the more practical approach is to require assignment of technology-specific derivative improvements and prohibit training, distillation, or competitive replication using escrowed assets, outputs, or traces.

Customers may reasonably ask why improvements they fund should not remain entirely theirs. One practical solution is a license-back structure that permits continued use of those improvements within the continuity deployment for the duration of the transition period, without conveying broader ownership rights.

Who pays the escrow agent and the successor operator, and which events beyond insolvency trigger release, such as acquisition by a competitor or discontinuation of the product, should also be settled at signing rather than after a trigger event.

Finally, few of these protections are likely to be actively monitored once the licensor is gone. Their value lies elsewhere. If assignment of improvements, use restrictions, and a defined end date are agreed in advance and made expressly transferable to the licensor's trustee or acquirer, an acquirer of the technology inherits clean title and greater freedom to operate, and the licensor or its estate can command a better price in a sale. A simple obligation on the successor operator to document improvements made during the continuity period gives that acquirer a clear inventory of what it owns.

The objective is not to prevent continuity. It is to ensure that continuity rights do not become a back-door mechanism for transferring ownership of future technology or control over the platform's roadmap.

A Short Drafting Checklist

For licensors negotiating AI continuity provisions:

- Select the continuity tier that makes sense, and architect the deployment so that tier is actually achievable
- Pre-approve successor operators and exclude competitors

- Set a hard end date for the release license
- Prohibit training, fine-tuning, or distillation from outputs and agent traces
- Define who pays and which events trigger release
- Make these rights expressly transferable to the licensor's trustee or acquirer
- Assign continuity-period improvements, and make the associated protections transferable, to the licensor and its successor in interest

The Realistic Goal

No continuity framework is perfectly enforceable.

The goal is not perfection. It is risk reduction.

Traditional software escrow assumes that maintenance boundaries are visible and reasonably enforceable. Agentic AI challenges both assumptions. The harness often contains the most valuable intellectual property. Maintenance activities can create new capabilities. Distillation can extract value without direct access to underlying assets. And many forms of enhancement are difficult to detect and classify.

For those reasons, continuity planning should focus less on what can be released and more on what must remain controlled, by whom, and for how long.

In agentic AI, continuity is ultimately not about handing over the crown jewels when a licensor fails or exits. It is about preserving service long enough to navigate a transition while protecting the ownership, control, and future value of the technology.

Notice: This content is provided for general informational purposes only and does not establish an attorney-client relationship or constitute legal advice. It may not be complete, accurate, or current, and you should seek guidance from a qualified attorney before taking any action. This content may have been organized with the assistance of artificial intelligence. The content may change without notice and could be considered Attorney Advertising in certain jurisdictions.