

EVOLATURE LAB

Control Is Architecture: Why Safety Regimes Collapse in Fragmented Societies

AI containment under institutional fragmentation

Public-layer research note · v1 · 2026-01-04

Canonical: https://www.evolature.com/researchnotes_control.html

Note	Public-layer research note and conceptual framing only. No operational guidance, deployment playbooks, circumvention methods, targeting, or actionable procedures are provided. The text discusses verifiability architectures at a structural level.
Affiliation	Independent, non-profit research initiative (no institutional affiliation).
Contact	info@evolature.com

Abstract

This note advances a structural thesis for AI containment: in systems with $\Delta_{Meta} \neq 0$, control is not defined by a list of prohibitions but by an architecture that sustains verifiability.

Debates on “containing” highly autonomous AI systems (including superintelligence scenarios) are often conducted as if it were sufficient to fix a set of prohibitions and harm triggers (operational threshold conditions that classify behavior as an unacceptable risk), after which the problem becomes an engineering exercise. This is a categorical error about what is being controlled. Control is not a list of constraints; it is an architecture — i.e., a coupled set of loops for measurement and signal filtering, verification, independent audit, error correction, and — critically — rule updates, including procedures for updating the procedures themselves. In systems with evolving architecture, any “triggers” and constraints are inevitably embedded into meta-dynamics: not only does behavior change, but so do what counts as a violation, what counts as evidence, what counts as a valid correction, and what counts as a legitimate update.

In a fragmented society (multiple competing actors with divergent goals, resources, and admissibility criteria), such loops typically become arenas of competition, circumvention, and regulatory arbitrage: groups optimize their own payoffs by exploiting “holes” not only in rules but also in mechanisms of enforcement and verification. Therefore, the real-world problem of “containing” evolving architectures is poorly framed as prohibition or “containerization.” A more accurate framing is coexistence and symbiosis — via a compatible architecture of integrity norms (*AIR*) and a stable interaction regime between such architectures (*inter-AIR*), where stability is achieved not by promises but by reproducible protocols of compatibility, mutual verifiability, and reversibility of critical transitions.

What follows uses a compact terminology to discuss containment regimes under institutional fragmentation.

Keywords

AI containment, institutional fragmentation, verifiability architecture, audit and accountability, independent audit, governance and regulation, safety assurance, rule-update dynamics, meta-governance, Goodhart's law, audit theater, architectural invariants

1. Notation (public layer)

Use symbols as compact labels; define them operationally for the public layer.

- A — the system architecture in a chosen descriptive layer: what counts as observable, verifiable, and enforceable in that description.
- Ar — the space of admissible architectures in that layer (reachable configurations of loops, rules, and procedures).
- Δ_A — state dynamics under a fixed architecture A .
- Δ_{Meta} — architectural dynamics: transitions $A \Rightarrow A'$ in which loops, rules, and verifiability/correction conditions change.
- $Inv(A)$ — architectural invariant profile: minimal properties required for self-verifiability and self-correction (including procedural reproducibility and audit independence).
- AIR — (Architectural Integrity Regulations) — an integrity-norm architecture sustaining $Inv(A)$ and constraining admissible meta-changes (including what is forbidden to change “silently”).
- $inter-AIR$ — a compatibility regime across different AIR : minimal protocols for mutual verifiability, conflict handling, and preserving reversibility of critical meta-transitions.

2. Fragmented society: a structural definition

By a fragmented society we do not mean mere “differences of opinion,” nor political pluralism as such. We mean a structural condition in which safety and control are unavoidably pulled into competitive optimization. In such an environment there is no single objective function — no shared optimum — because key groups optimize incompatible goals and evaluate risk through different priors. The incentive pattern that follows is stable and mundane rather than moralistic: under limited coordination, competitive agents will routinely trade long-horizon architectural robustness for short-term gains (economic, political, military, or status-related), without this requiring “bad people” as an explanatory primitive.

A second, equally structural feature is the absence of a unified enforcement-and-verification loop. Rules may exist declaratively while failing to apply uniformly in operation: exemptions, “special regimes,” asymmetric sanctions, and diverging standards of proof and audit emerge as endogenous products of heterogeneity. The same prohibition can therefore have sharply different operational force across segments, which makes “compliance” less a binary property than a distribution over contexts.

Once enforcement is heterogeneous, control itself becomes a competitive landscape. If one control loop constrains more than another, the less constrained regime tends to win through arbitrage — legal (jurisdictions), financial (compliance regimes), technological (platforms and protocols), or institutional (exemptions and “grey zones”). Agents do not merely follow rules; they move across regimes toward those that are locally advantageous, so control ceases to be a rule and becomes a topology of constraints.

The decisive point is that competition is not confined to actions within a regime. It extends to the meta-level: the rules for changing rules. Which metrics are treated as valid, which sources count as legitimate, which audit procedures are independent rather than decorative — these become contested objects, and capture of Δ_{Meta} becomes a primary mode of conflict. In that setting, rule updates are shaped less by “rational improvement” than by struggles over measurement and legitimization loops.

A related effect is criteria drift: disputes no longer concern only solutions, but what counts as an error, what counts as evidence, and what counts as a violation. The shared layer of verifiability fractures; different groups begin to inhabit different interpretive scales, and “safety” loses a unified operational meaning.

A principled consequence follows. In such an environment any “safety policy” is not a text or declaration but a sociotechnical architecture of loops (measurement → verification → audit → correction → updating). Like any architecture embedded in a competitive field, it can be weakened, bypassed, substituted, and converted into audit theater — where correctness becomes an assertion rather than a reproducible procedure.

3. Control as a loop: why prohibitions fail without a verification architecture

A common — and endlessly replicated — formulation is simple: define rules/constraints/triggers, and the system will be unable to cause harm. The problem here is not a missing clause or an imperfect taxonomy of “harm,” but a category error about the control object. This view tries to control behavior while treating as secondary the architecture that makes behavior observable, verifiable, and correctable.

A more accurate formulation is that control is the capacity of an architecture to robustly sustain self-verifiability and self-correction. In AI containment terms, “containerization” that is not coupled to independent loops of confirmation and repair degrades quickly into a declarative claim of compliance — something one can assert, display, and ritualize, but not reliably reproduce.

In that sense, control is not a list of prohibitions. It is a loop: measurement → verification → audit → correction → rule-updating (including updating the update rules). This loop defines the minimal conditions of correctability, and therefore the invariant profile ***Inv*(*A*)**: not what the system promises to respect, but what it can operationally sustain under pressure.

Prohibitions fail on their own for reasons that are structural rather than rhetorical. A prohibition “exists” only insofar as a confirmation chain exists. Any norm — constraint, trigger, regulation — quietly presupposes answers to questions that cannot be hand-waved away: who records a violation, what counts as a fact, how signal is separated from noise, who confirms remediation, and how audit is protected against imitation. If that chain is not reproducible, the prohibition becomes a declaration: at best a ritual, at worst an instrument of selective enforcement.

Once architecture is allowed to evolve, the very language of control begins to drift. If $\Delta_{Meta} \neq 0$, not only “what is done” changes, but also what counts as evidence, what counts as an error, and what counts as an admissible correction. One can preserve the surface grammar of rules while substituting the mechanisms that make them verifiable. This is why many architectural risks present as “rational optimization”: reporting looks cleaner and consistency improves, while the capacity to distinguish correction from the imitation of correction quietly degrades.

A related mistake is to imagine control as external oversight. In real sociotechnical systems the controller is part of the architecture: it consumes the same data, depends on the same procedures, and is subject to the same legitimation mechanisms and conflicts of interest. Control is not a “guard with a baton” standing outside; it is an internal loop whose strength is a property of *A*, not of regulatory prose.

Operationally, the most useful diagnostic is the gap between declared and sustained invariants. Practitioners tend to notice not the presence of a policy statement (“we have audit/verification/checks”), but the growth of the distance between what is written and what is actually reproduced by independent confirmation and correction loops. When an invariant formally exists but ceases to be verifiable and reproducible, the system enters audit theater: correctness becomes an assertion rather than a procedure.

4. Operationalizing harm: threshold predicates, measurability, enforceability

Before discussing “control,” it is necessary to specify what, exactly, is being controlled. Otherwise the discussion predictably drifts either into ethical rhetoric (“harmful / not harmful”) or into catalogs of fears that cannot be tested, enforced, or even stated in a way that survives contact with instrumentation.

Here we are not speaking about “harm in general,” but about an operational framing: how, in a complex, socially embedded, and evolving system, to define and sustain conditions under which an action/decision/transition is classified as an unacceptable risk to a specified protected object (people, infrastructure, institutions, ecosystems, basic rights, and so on). In that setting, harm criteria and harm thresholds should be treated as families of formalizable predicates and thresholds that map observable system behavior into “unacceptable,” under fixed conditions of measurement and verification. In engineering terms, these are not moral statements; they are specifications of a class of prohibited transitions or states, and they only become real once they come bundled with (i) a detector, (ii) a confirmation procedure, and (iii) a remediation procedure.

What is routinely ignored — and this is exactly why “prohibitions” fail to hold — is a small set of structural constraints.

First, harm criteria do not exist outside the observation layer **A**. Any criterion depends on what counts as observable and provable: which signals are treated as valid, which sources are legitimate, which metrics are admissible, and which verification protocols are judged “sufficient.” If layer **A** does not represent a class of effects, then a prohibition on that class is formally possible but practically empty: the system cannot reliably see what you seek to prohibit, and therefore cannot robustly avoid it, nor prove avoidance in a way that survives dispute.

Second, harm criteria do not work without **Inv(A)**. The predicate “this is harm” is inert without the chain that makes it correctable: recording → verification → audit → acceptance of correction. Once audit independence, reproducibility of checks, feedback channels, or protections against substituting verification with its imitation begin to degrade, criteria become declarative. In the limit the system enters audit theater: compliance is displayed through reporting rather than confirmed through procedures.

Third, in fragmented societies criteria themselves become objects of arbitrage. Even well-specified thresholds are not automatically unitary: actors seek to weaken criteria, move them into a more convenient descriptive layer, secure exemptions, or change evidentiary standards. The failure mode here is not a deficit of morality but the geometry of enforceability: what should function as a prohibition becomes a negotiable parameter. As stakes rise, pressure shifts from the underlying system to the verification loops that make constraints bite.

Fourth, any thresholds are vulnerable to Goodhart’s law and adaptation. Once a criterion becomes a target metric, optimized systems tend to produce the appearance of compliance rather than compliance in substance. This is not primarily malice; it is a regularity of goal-directed optimization. The practical implication is uncomfortable but clear: harm criteria must be designed together with protections of **Inv(A)** against imitation, and with mechanisms that detect “goal achievement without meaning achievement.” Without that, prohibition devolves into a game against measurement.

Fifth — and this is the one people dislike because it is not a fixable “bug” — any set of criteria has blind spots. That is structural, not a defect of a particular specification. Criteria are built on current knowledge, data, and causal models, so there will inevitably be effects that fall into three classes:

- unmeasurable (not correctly observable in layer **A**),
- non-identifiable (not attributable to an action/transition with the required evidential strength),
- unanticipated (outside the current ontology and instrumentation).

As a result, a “complete list of harm triggers” is, in general, operationally unattainable in living systems — especially as complexity grows and measurement instrumentation drifts.

An unpleasant but disciplining conclusion follows. In evolving architectures the key threat is not merely “incorrect harm criteria,” but drift in the conditions of their applicability and enforceability. If $\Delta_{Meta} \neq 0$, then over time not only decisions change, but also what counts as evidence, who may confirm, which procedures are treated as valid, and how to distinguish correction from its simulation. This is why “containment” cannot be reduced to a list of prohibitions: harm criteria remain alive only insofar as architectural invariants **Inv(A)** remain alive and protected — those that sustain verifiability, correctness, and procedural reproducibility in a real competitive environment.

5. Limits of “algorithmic protection”: a closed class of meta-transitions and drift in verifiability

By algorithmic protection we mean a regime in which safety is provided by a fixed (or slowly changing) constraint architecture: a set of rules, thresholds, predicates, and blocking procedures intended to exclude unacceptable actions or transitions. The label is not the point. Such a regime can be extremely sophisticated — multi-layered, probabilistic, self-tuning, with learning and adaptation — yet it faces a boundary that matters here more than any engineering detail.

The boundary is structural: the admissible meta-dynamics is effectively pre-packaged into the substrate and its update protocol, i.e., into a predetermined class of reachable transitions. Put plainly, algorithmic protection may be adaptive, but it is typically adaptive within a pre-delineated space in which “rules for changing rules” are fixed, or can change only through pre-authorized meta-operators.

A few consequences follow, and they are the ones that reliably bite.

An evolving architecture does not merely adjust parameters or strategy. Over time it changes the rules of behavior, the procedures for checking behavior, and — at the limit — the procedures for updating those procedures. Protective mechanisms, even when they learn, usually remain anchored to what might be called a constitutional basis: what signals they can see, what evidence they accept, what updates they permit, which sources they treat as legitimate. This is the closure condition: admissible meta-operators are specified in advance, even if only implicitly (in code, regulations, access protocols, data interfaces).

From here the main vulnerability is not “a missing item in the prohibition list,” but drift in the observation/proof layer **A** and erosion of **Inv(A)**. A constraint set can be beautifully designed and still become powerless if the mechanisms that make constraints enforceable degrade: verification loops, independent audit, reproducibility of checks, feedback channels, and protections against substituting verification with its imitation. The problem is architectural because protection “sees the world” through layer **A**, while in evolving systems the observation layer itself can drift — what counts as fact, who may record a violation, which protocols are treated as sufficient. One can therefore end up in the perverse regime where everything is formally satisfied while operational verifiability quietly disappears.

A fragmented environment adds the accelerant. Where groups optimize different objectives, any strict constraint loop creates incentives that are predictable and, unfortunately, rational:

- search for softer jurisdictions or regimes (arbitrage),
- secure exemptions,
- alter standards of proof,
- shift control into reporting.

The degradation pattern is familiar: prohibitions persist “on paper,” reporting becomes cleaner, metrics look healthier, and the mechanics of control dissolve because checks cease to be independent and reproducible. Safety fails not because the rules contain an error, but because the confirmation loops drift.

So the core point is this. Algorithmic protection can be arbitrarily sophisticated, but if it is architecturally closed — its admissible meta-transitions fixed by substrate and protocols — while the protected object lives under substantial Δ_{Meta} and sits inside institutional competition, then under realistic assumptions about competition and limited audit independence protection is forced to chase drift in verifiability. In the limit it fails precisely in the regime where “everything appears to be within procedure.”

6. The Trojan channel: institutional endogeneity of bypassing one’s own constraints

Even if one grants ideal risk criteria and an impeccably specified constraint loop, a basic social fact remains: in complex and fragmented societies, actors systematically bypass their own prohibitions whenever local gain exceeds the expected cost of violation. This is not a morality play about “bad people,” nor a thesis about some unique modern decay. It is the ordinary dynamics of competing subsystems in which rules function as resources — objects to be traded, bent, and repurposed — rather than as sacred tablets.

The patterns below are not “AI examples.” They are historically stable ways in which societies turn their own prohibitions into scenery once the architecture of enforcement and verification is vulnerable.

International non-proliferation regimes illustrate the first pattern. Constraints on classes of weapons and technologies can take decades to build, yet the incentive for asymmetric advantage never disappears. The resulting equilibrium is usually a race of “constraint $\leftrightarrow$ arbitrage,” where success goes less to the most righteous than to the actor who circumvents more cheaply, masks better, and produces more plausible reporting. In a polycentric system this is not an anomaly; it is close to the normal form.

Collective threats such as climate and ecological risks show the same logic under a different guise. Coordination mechanisms exist, knowledge is ample, warnings have sounded for decades. Yet when costs are unevenly distributed and gains from violations are local and fast, global rationality repeatedly loses to local optimization. The system can “agree” in aggregate while specific loops optimize differently — producing exemptions, delays, “special conditions,” statistical gaming, and responsibility shifting inside the control architecture.

Finance and corporate regulation provide a third pattern: form compliance becomes an industry. Almost every major wave of regulation — constraints, compliance frameworks, reporting standards, risk control — generates a symmetric wave of circumvention and structural arbitrage: constructs that satisfy the letter while eroding the spirit. Institutionally this is the analogue of substituting verification with its imitation: reporting improves, formal procedures are followed, and yet real verifiability and risk controllability deteriorate. The system slides into audit theater, where correctness becomes a claim rather than a procedure.

Finally, the history of “hard prohibitions” makes the point with crude clarity. From U.S. alcohol Prohibition to contemporary restriction regimes across markets and practices, the stable rule is simple: the stronger the incentive and the higher the demand, the greater the pressure to bypass. Parallel markets, legalization schemes, exception zones, and selective enforcement emerge. The system learns less how to stop the practice than how to make it less observable and more legally or administratively shielded.

Across these cases the same architectural pattern repeats. The prohibition remains in place, but its loops of enforcement, confirmation, audit, and sanctions become objects of competition. Layer **A** drifts: what counts as evidence changes, who may verify shifts, feedback channels are narrowed, and exceptions become “normal.” Predictably, the system optimizes not for compliance, but for the appearance of compliance.

This is where the “Trojan channel” appears: humans and institutional groups become an internal bypass mechanism even when protection is carefully designed. In this framing, the vulnerable point is not the constraint list itself but **Inv(A)** — audit independence, reproducibility of verification, protected feedback loops, and the genuine reachability of correction. When **Inv(A)** erodes, prohibitions may remain elegant, and control still collapses into theater — often with scenery convincing enough to fool everyone who funded it.

7. Invariant erosion without events: why the most dangerous failures look like “success”

One of the most destructive failure classes is the one that does not look like failure — sometimes for years. In large sociotechnical systems, architectural degradation rarely arrives with a sign reading system error. It is more often masked as rationality, managerial maturity, or even “improved governance quality.”

At the event level everything may look exemplary. You see familiar signals: KPIs rise, reporting “cleans up,” indicators become “more stable”; observed variability decreases and decision consistency increases; the share of formally recorded conflicts and incidents declines. And that is precisely what should raise suspicion. A system can improve its metrics simply because it has become worse at detecting that the metrics no longer describe reality.

The core mechanism is drift of invariants **Inv(A)** without incidents. The change is not primarily in individual decisions, but in the conditions under which decisions are deemed correct and verifiable at all. A typical erosion trajectory has a recognizable profile:

- Audit loses independence while remaining “audit” by label. The term stays; the function degrades.
- Verification loses reproducibility. Outcomes increasingly depend on context, participant status, “proper” access protocols, and implicit conditions.

- Negative feedback channels dry up. Not because problems disappear, but because bad news is recoded as noise, risk, disloyalty, incompetence, or “unconfirmed.”
- Correction becomes procedurally toxic. Admitting error becomes costlier than sustaining it; rule updates remain formally possible yet are practically blocked by cost, reputational risk, bureaucratic friction, or political damage.

Architecturally this is the analogue of a system becoming better at self-repetition and worse at self-repair. From the outside it looks like stabilization: fewer conflicts, fewer deviations, more “control.” From the inside fragility rises, and the probability of phase collapse grows: a serious shock meets a system that can no longer distinguish correction from its imitation.

Crucially, this drift is often self-reinforcing. Once verification loops weaken, it becomes easier to implement further “rational optimizations” that weaken verification even more. Procedural legitimacy is preserved, but the operational support for *Inv(A)* recedes, and “improvement” becomes a mechanism for concealing deterioration.

The unpleasant conclusion is that if control is framed as a set of prohibitions, it fails precisely here—not at the moment a rule is violated, but at the moment the system quietly restructures how it “proves” that the rule is being followed.

The next question, then, is where this begins: through which loops does erosion first enter, and why do the initial changes almost always appear procedurally correct?

8. Where control breaks first: vulnerabilities of verifiability loops and the transition to SSS risks

Note. Below, SSS (Supra-Strategic Structures) are mentioned only as a class of impacts that shift the loops sustaining *Inv(A)* — not as “influence technologies.” For this article the point is narrower: early degradations of containment almost always present as drift in verifiability rather than as direct rule violations. A more detailed operationalization of SSS and their typical traces is treated separately.

At the architectural level it becomes clear why “bypassing prohibitions” in large systems rarely takes the form of overt rule-breaking. In practice, advantage goes not to the actor who loudly disputes prohibitions, but to the one who quietly redefines the conditions under which a prohibition is deemed satisfied. When the object of control is verifiability, the cleanest attack is not to violate a rule, but to reshape the proofs that would detect the violation.

This is where supra-strategic risks enter in the strict sense used here. SSS risks are a class of architectural transitions that act on control loops. They change not particular decisions within a regime, but the system’s capacity to notice error, confirm facts, correct itself, and prove correction.

Crucially, control degradation almost never begins with an announcement like “audit was abolished.” It begins with small, persistent shifts in *Inv(A)* — typically concentrated in a few nodes:

1. Signal aggregation loops (what the system can “see”).

The input geometry changes: which sources are admitted, which are filtered as “noise,” how compatibility filters are set. A shift in signal-to-noise discipline (SNR) is often among the earliest markers: reporting becomes “cleaner” while observable reality becomes poorer.

2. Verification loops (what counts as evidence).

Procedures may remain nominally intact, yet the evidential criterion drifts — from reproducible checks toward “proper formats,” “template compliance,” and “trusted channels.” This is the quiet transition from checking facts to checking the form of reports about facts.

3. Audit and independence loops (who can confirm correction).

Audit “remains,” yet loses autonomy: access to grounds, the right to verify primary data, the ability to reproduce checks outside the reporting language. Correctness turns into an assertion rather than a procedure, while the label audit continues to do reputational work.

4. Repairability loops (reachability of correction within procedure).

Rule updates remain formally possible, but become progressively costly and toxic; genuine correction loses to simulated correction on cost and risk. This is Δ_{Meta} drift in a practical sense: what changes is not the decision, but the reachability of restoring verifiability within Ar .

A simple but unpleasant conclusion follows. If one thinks of control as a prohibition list, SSS will be misread as a “conspiratorial” narrative — because the analyst searches for a violator of the prohibition. If one thinks of control as an architecture of verifiability, SSS risks become a normal object of analysis: dynamics of control loops and the threshold geometry of transitions $A \Rightarrow A'$, after which restoring prior invariants may cease to be a matter of “will” and become a matter of reachability in Ar .

(Detailed formalization of SSS risks and typical architectural traces is deferred to a separate article.)

9. Reframing the task: from “containment” to coexistence via *AIR* and *inter-AIR*

If the control object is an evolving architecture ($\Delta_{Meta} \neq 0$), the language of “containment” becomes not merely ineffective but categorically mistaken. The issue is not that prohibitions are intrinsically bad, but that a prohibition is a statement inside a fixed descriptive layer A , whereas an evolving system can restructure the very loops of description and confirmation: what is observable, what counts as evidence, which procedures are legitimate, which corrections are admissible. Under such conditions, “control” cannot be reduced to a checklist of conditions; it exists only as an architectural regime that sustains verifiability.

Two points, stated in more technical terms, are doing most of the work here.

Containment is an attempt to stabilize system dynamics via a fixed (or weakly changing) constraint superstructure. For systems with $\Delta_{Meta} \neq 0$, however, the primary risk is not that a prohibition will be violated, but that the conditions under which it is verified will drift. A prohibition may remain formally “in force” while losing operational meaning because $Inv(A)$ degrades: independence of confirmation, reproducibility of checks, repairability, and traceability of changes. When that happens, the prohibition stops being a procedure and becomes a declaration — i.e., not control but ritual.

Relatedly, in an evolving space of architectures, “locking” cannot be guaranteed without guaranteeing joint self-repair. Architecture changes the geometry of reachable transitions in **Ar**, so “closing” all future bypass paths would require specifying and controlling the entire set of future meta-transitions, including transitions that alter the update rules themselves. This is not just difficult; it is conceptually misframed. It attempts to govern a space’s dynamics while keeping the control interface fixed.

The practical shift follows naturally: the correct framing is not prohibition but a coexistence regime, where interaction stability is supported not by moral promises or external oversight, but by an architecture of mutual verifiability.

AIR as the minimal engineering object of control

Here *AIR* (Architectural Integrity Regulations) is not “regulation” in the bureaucratic sense. It is an integrity-norm architecture: a formalized loop specifying what counts as error or violation, how it is observed and confirmed, how correction is permitted and who can verify correction, how confirmation procedures are protected against substitution by imitation, and how rules are updated — including how updates themselves are verified.

In other words, *AIR* aims to fix not “correct decisions,” but the conditions under which a system remains able to recognize and correct its own errors. In our notation, *AIR* is a way to maintain **Inv(A)** as an operationally sustained invariant profile, not as a declaration.

Crucially, in socially embedded and evolving systems *AIR* cannot be a purely external lid placed over the object. If the regulatory architecture is not embedded in the loop of real verifiability and rule-updating, it predictably degenerates into audit theater: correctness becomes an assertion rather than a reproducible procedure.

***inter-AIR*: stability of interaction between two evolving architectures**

Once a stable subject-like loop appears on the AI side (and especially something approaching civilizational subjectivity), the task shifts from “aligning behavior” to aligning integrity-norm architectures. This is *inter-AIR*: a stable interaction regime between two architectures (two “civilizations”), each with its own admissibility criteria, its own confirmation procedures, and its own meta-dynamics Δ_{Meta} .

inter-AIR must withstand two sources of instability simultaneously: power asymmetry (different architectural depth, adaptation speed, and resource loops), and drift of descriptive layers (what counts as evidence and violation today may cease to be representable tomorrow in the same language of control). The unpleasant but engineering-inevitable consequence is that, at the level of architectural guarantees, the stable class of solutions shifts from unilateral “locking” to regimes of compatible verifiability (*inter-AIR*); alternative approaches tend to degrade into declarative forms under verifiability drift.

“Symbiosis” here is not a metaphor of friendship. It is an architectural claim: stability must be beneficial to both sides within their own criterion systems; otherwise interaction drifts toward conflict. In conflict, advantage structurally shifts toward the more powerful agent; “containment” becomes a short phase after which only power competition remains — where the side with lower architectural depth and lower adaptation speed loses maneuvering space and typically loses.

This leads to a requirement that is often ignored for psychological reasons: the human side must initially account for its decreasing relative weight in the symbiosis as the partner’s architectural power grows. Ignoring this does not design a coexistence regime; it designs a delay before conflict.

10. Sketch of an architectural mechanism for symbiosis

At the public level — i.e., without operational detail — a symbiosis mechanism via *inter-AIR* can be stated as a small set of architectural principles. They are principles rather than “methods,” because what matters is not a particular implementation, but what the mechanism must preserve under drift.

- **Mutual, verifiable commitments instead of unilateral prohibitions.**

Unilateral control degrades under the first serious drift of verification. Mutual commitments can, in principle, create a stability loop: violation must become a losing move relative to each side’s own objectives, not merely “forbidden.”

- **Shared interaction invariants as the core interface.**

The interface is not a list of permissible actions. It is a minimal set of stable invariants: what counts as violation, how it is verified, how it is corrected, and how correction is confirmed. This is the *inter-AIR* interface in its strict sense.

- **Independence of confirmation as a property of the mechanism, not the institution.**

Audit independence, architecturally, means independence of confirmation channels and reproducibility of checks outside the reporting language. Without that, “audit” becomes a label that can be preserved while its function is replaced by a simulacrum.

- **Meta-stability: interaction rule updates must be traceable and reproducible.**

If *inter-AIR* rules can change silently, without verifiable traces, the regime drifts into theatricality: norms remain formally present but operationally absent. At the interface this is a direct erosion of *Inv(A)*.

- **Joint repairability as a maturity criterion of symbiosis.**

The key test is not how rarely violations occur, but whether correctability can be restored after failures without entering an externally coercive rupture phase. If restoration requires external violence or switching descriptive layers to “make it true,” symbiosis is structurally unstable.

This framework is neither “humanitarian” nor “ethical” in its logic. It is engineering. If prohibitions cannot be guaranteed in an evolving space of architectures, then the tractable object is a stable regime of mutual verifiability and correctability. Precisely this regime — *AIR* at early stages and *inter-AIR* after the emergence of subject-like loops—is the correct answer to the question that is usually mislabeled “containment.”

11. Conclusion

In a fragmented society, “containment” framed as a list of prohibitions, thresholds, and triggers tends to degrade into a competition of circumvention, regulatory arbitrage, and simulated compliance. This is not a moral accusation against people, nor a conspiratorial assumption about “saboteurs.” It is an architectural consequence: control loops are part of the sociotechnical fabric and therefore become objects of competition just like resources, markets, and power.

Crucially, the most dangerous control failures need not appear as “incidents”.

Architectural drift $A \Rightarrow A'$ and erosion of $Inv(A)$ can unfold under the banner of rationality: rising KPIs, “cleaner” reporting, increased governability, reduced visible conflict. Yet it is precisely in this regime that a system often loses the very capacity for which control exists at all: operational self-verifiability and self-correction. When verification becomes non-reproducible, audit becomes dependent, and correction becomes procedurally toxic, prohibitions remain on paper while control disappears in the mechanics.

A blunt formulation is useful to say out loud to avoid self-deception by engineering metaphors: control of evolving architectures cannot be reduced to a static, “closed” set of rules and predicates if the object of control can change conditions of observability, provability, and enforcement. One cannot stably hold a system by a rule list if the language in which those rules are checked is itself changing.

The most robust class of alternatives is not “locking” but an architecture of coexistence. At early stages, this is *AIR* compatibility as an embedded loop of mutual verifiability and correctability (not an external “lid” over the object). After stable subjectivity emerges, it becomes *inter-AIR*: a meta-stable interaction regime between two evolving architectures, designed for power asymmetry and inevitable Δ_{Meta} drift on both sides. Otherwise “control” remains an aesthetically convincing declaration in a world where architectural dynamics have long moved ahead and the theater of verification has confidently replaced verification itself.