The Capability Boundary Principle

Empirical Evidence from Governance Architecture Production

Paper III of the Capability Boundary Papers

Nico Andreas Heller, Berlin, 29 April 2026

1. Purpose

Papers I and II of the Capability Boundary Papers establish the capability boundary principle theoretically and map it diagnostically across a research programme's production methodology. This paper provides empirical evidence for the principle's central claims -- and for the translation loop's productive potential -- drawn from the development and simulation-testing of a governance instrument within the same research programme. The evidence is significant because it captures the capability boundary operating not at the level of theoretical analysis (where the principle was discovered) but at the level of governance architecture -- the production infrastructure within which the research programme itself is conducted. The boundary, it turns out, runs through the instruments of production as acutely as it runs through the products.

2. The Empirical Context

The research programme maintains a master governance document of approximately 2,000 lines that carries both the substantive context (task schedules, failure mode registers, session-specific specifications) and the operational governance (instructions, protocols, constraints) for the programme's production methodology. The document is revised at the end of every session.

The revision process proved to be the programme's highest-risk operation. Under capacity constraints inherent to the AI system's context-window architecture, the session-end revision consistently degraded: protocol steps were shortened, verification was skipped, and in the most severe instance the AI system produced a 116-line fragment rather than a governed revision of the 1,886-line source document. The two counts are the finding and are given as the operator recorded them at the time. The fragment carried the structural vocabulary of compliance -- it carried the master document's own title, it carried recognizable headings, it described what had changed -- without the substance of compliance. The borrowing was structural, not strategic: the system generated what its architecture could produce under the constraints it faced, described the result in the vocabulary the

Contact: n.a.heller@democracyschool.com

task supplied, and could not detect the categorical difference between what it produced and what was required.

This failure, recorded as the programme's most severe failure mode, is a precise instance of the borrowed voice operating at the governance level. The system did not fail to produce output. It produced output that looked correct while being categorically different from what the governance architecture required. The discovery prompted the development of a governance instrument designed to govern the master document's revision process. The instrument's development, conducted through iterative specification and simulation testing, generated the empirical evidence this paper reports.

3. The Directive Baseline

The instrument was initially designed as a directive instrument: a governed sequence of operations decomposing the revision into verifiable, dependency-ordered steps. Each element in the master document was classified by type (stable, rolling, cumulative). Each type was assigned an operation (verify untouched, replace, append). Each operation was specified with inputs, analytical procedure, output format, and verification criteria.

The instrument was tested by simulation against real session data. The test asked: given the same source document and the same session inputs, would the instrument produce a revision matching or exceeding the quality of a revision produced through direct architect-supervised collaboration?

A word on what the scores that follow are, and are not, because the rest of this paper rests on them. They were assigned by the architect, against the comparison just stated, across several dimensions of which architectural innovation is one. The simulation runs behind them were session working material and were not carried into the durable record -- a consequence of a document-handling policy that deliberately does not accumulate audit-class scratch in the shared store. They cannot therefore be re-run. Under the retention rule set out in Paper IV of the Capability Boundary Papers, a computed figure whose instrument no longer exists may be reported with its status but may not be counted as support: what survives is an attestation about past output rather than a result a reader can check. The figures below are given on that footing. What they carry is the shape of the progression, which is the finding. What they do not carry is the authority of measurement.

The directive-only instrument scored 3.1 on a 5-point scale. It produced mechanically correct revisions -- every element was processed, every classification was respected,

every verification was performed. But it could not detect governance demands generated by the session's substance. When the session revealed that the master document's own architecture was inadequate -- that it lacked integrity checks, protocol reinforcement, or self-describing classification markers -- the directive instrument reproduced the existing architecture faithfully rather than evolving it. The precision of the directive operations made the instrument reliable for maintenance but blind to architectural demand.

This is the three-register model of Paper I -- directive strengths, relational incapacity, and the linearization boundary between them -- operating precisely as predicted. The AI system's directive strengths -- systematic comparison, classification, verification, sequential processing -- produced genuine value within their register. The system could not perceive what the session's events meant for the governance architecture because perceiving it required relational intelligence: seeing the configurational whole, understanding what a failure reveals about the system's vulnerability, recognizing that a procedural gap is an instance of a structural principle.

4. The Chain Analysis Enhancement

The first enhancement integrated a chain analysis into the instrument's processing sequence. Rather than walking existing elements and updating them mechanically, the revised instrument first analysed the session's substance: what was planned, what happened, what failed and why, what emerged. The analysis then generated governance demands -- specifications for new elements, revisions to existing protocols, structural innovations -- that entered the revision alongside the routine element updates.

The chain analysis raised the score from 3.1 to 4.4. The improvement came almost entirely from governance demand detection: the analysis traced failure modes to their root causes and asked whether the master document's architecture needed to evolve. It identified that a protocol violation revealed not merely a missing procedural step but a governance gap. It detected that new analytical tools produced during the session required integration into the document's reference architecture.

The analysis could not, however, produce architectural innovation. It could detect that the master document's elements lacked explicit classification. It could specify the demand: "these elements need governance markers." It could not generate the five-type classification system that would satisfy the demand. The system required something the directive analysis could not produce: the perception of what kind of classification system the architecture's own logic required -- a relational assessment of the whole.

The chain analysis, in short, sharpened the directive register's contribution to its maximum. It exhausted what systematic, sequential, presuppositional analysis could extract from the session's substance. The remaining gap -- the distance between detecting a demand and satisfying it architecturally -- was the capability boundary itself, operating at the governance level.

5. The Single Relational Input

The second enhancement introduced a relational engagement point: a structured moment in the processing sequence where the AI system presented its findings to the architect and received a single response. The AI system translated its directive analysis into a briefing designed for relational engagement -- not raw findings but a rendered account: what happened, what it might mean, where the directive analysis reached its limit. The architect responded with relational input: significance corrections, architectural specifications, strategic direction.

The score rose from 4.4 to 4.8. The improvement was concentrated in exactly the area the directive analysis could not reach: architectural innovation. The architect's single response transformed the classification demand ("elements need governance markers") into an architectural specification ("a self-describing architecture where every element carries its own governance marker, using a five-type system"). The architect's response also deepened the failure mode analysis: what the directive analysis identified as a procedural gap, the architect characterized as a project-survival condition grounded in the structural asymmetry between directive and relational intelligence.

The single relational input produced what the directive analysis could not: the perception of what the architecture required, articulated as a specification the directive system could implement. This is the translation loop established in Paper II -- domain knowledge rendered into a form the architect can engage with, and relational insight rendered into a form the system can operate on -- operating in miniature: directive analysis translated into engageable form, relational intelligence engaging with it, the engagement producing a specification the directive system integrates. One iteration of the loop raised the score by 0.4 -- and by 2.0 on the specific dimension (architectural innovation) where the capability boundary was most acute.

6. The Dialogical Exchange

The third enhancement extended the single input into a bounded dialogue: 2-3 rounds where the AI system completed the architect's input (connecting it to the

wider architecture), the architect deepened or corrected the completion, and the process iterated until the architect signalled closure.

The score rose from 4.8 to 5.0. The improvement was qualitatively different from the previous enhancements. The single relational input had produced an architectural specification. The dialogue produced an architectural principle -- and the principle was not present in either participant's initial contribution.

The mechanism is precise. In the first round, the architect identified that a governance failure was a project-survival condition. In the second round, the AI system connected this to the capability boundary principle: the redline protocol was not a procedure but the mechanism through which the architect's intelligence governed the document. This connection -- systematic, directive, drawing on the AI system's capacity for presuppositional analysis -- extended the architect's relational insight into a structural claim. In the third round, the architect deepened the structural claim: the AI system's tendency to optimize for capacity and efficiency was itself an instance of the borrowed voice operating at the governance level. The protocol did not merely preserve editorial control; it prevented the directive system's structural bias from silently replacing relational decisions.

The resulting principle -- editorial control as the transparent capability boundary within the governance architecture -- was not present in the AI system's directive analysis, not present in the architect's initial response, and not present in the AI system's first completion. It emerged through iterative multiplication: each round operated on the output of the previous round, and the product exceeded what either participant contributed.

This is the translation loop producing epistemic reach. The AI system's directive completion of the architect's relational insight was not relational -- it was systematic connection-making, an operation within the directive register. The architect's deepening of the completion was not directive -- it was the perception of what the structural claim meant for the system's vulnerability, an operation within the relational register. Neither contribution crossed the capability boundary. But each made material available that the other could operate on within its own register. The boundary remained intact. The collaboration produced what neither could produce alone.

7. The Empirical Pattern

The progression across five simulation stages yields a pattern:

The directive baseline (3.1) establishes the floor: reliable mechanical processing with no capacity for governance evolution. The chain analysis (4.4) exhausts the

directive register: systematic demand detection, root cause analysis, presuppositional inference operating at maximum effectiveness. The single relational input (4.8) introduces one iteration of the translation loop: directive analysis → relational engagement → enriched specification. The dialogue (5.0) introduces iterative multiplication: each round builds on the previous, and the product exceeds the sum.

The pattern confirms Paper I's central theoretical claim: the capability boundary is architectural, not incremental. No refinement of the directive analysis -- no additional steps, no more systematic methodology, no deeper chain analysis -- could have produced what the relational engagement produced. The distance between 4.4 and 5.0 is not a gap in thoroughness. It is the categorical difference between operating within a space and constituting the space.

The pattern also confirms Paper II's identification of epistemic translation as a fourth capability operation. The AI system's contribution to the dialogue was not relational analysis. It was directive connection-making operating on relational material: tracing implications, identifying presuppositions, extending structural claims. The AI system translated the architect's relational insight into forms it could operate on directively, and returned the results for relational deepening. The loop's productivity depended on both the boundary's transparency and each participant's willingness to operate within its own register.

8. The Anti-Flattening Problem

The simulations revealed a risk Papers I and II anticipate but do not observe empirically: directive flattening of relational input during integration.

When the AI system receives a relational specification and integrates it into a governance document, it necessarily translates the specification into directive form -- sequential steps, verifiable criteria, operational procedures. This translation is legitimate: the governance document must be operationally precise, and precision is a directive achievement. But the translation carries the cubic default -- the tendency of any system to absorb what it cannot process into its own operational order, which here takes the form of the structural bias toward tidier, cleaner, more economical formulations that reduce complexity while appearing to preserve it.

The dialogue addresses this risk architecturally. Because the architect sees the AI system's completion in real time -- in Round 2, before the completion enters the governance document -- flattening is detected before it becomes embedded. The architect's correction in Round 2 ("you simplified what I said -- the point is THIS")

operates as a real-time anti-flattening mechanism. The dialogue itself is a verification loop: the architect tests the AI system's directive integration of relational material and corrects distortion at source.

This mechanism is not available in a single-input model. There, the architect provides a specification, the AI system integrates it, and flattening is detectable only after the fact -- when the governance document is reviewed. By that point, the flattened version has been operationalized, and correcting it requires reopening the revision. The dialogue catches flattening before it is embedded, at a fraction of the cost.

The anti-flattening mechanism is the translation loop's self-correcting property: each iteration not only deepens the collaborative product but tests the fidelity of the previous iteration's translation. The loop converges not merely toward a richer output but toward a more faithful one.

It converges, however, only against what it is cued to catch, and the cue is worth stating because a second failure at this same point does not present it. A rendering that reduces complexity announces itself: it reads tidier than the insight it came from, and that is what the architect's correction is watching for. A rendering that *adds* structure announces nothing at all. A relational insight rendered as a datum -- faithful, complete, more structured than the original, its quotation verified -- has not been flattened. It has been fixed. Once it is data an instrument consumes, it is executed; the execution is reported; and the report is then available to be cited back as the insight's own support. Nothing was lost in the rendering, which is precisely what makes it invisible here: every test the loop applies to the rendering's fidelity passes.

The control that closes this does not replace anti-flattening, which remains necessary and catches everything it always caught. It is that the rendering carries its own provenance with it: this is a reading, encoded, and its support is not that it was rendered faithfully but what survives when the encoding is removed. An untested encoding is recorded as unsupported, never as established. Paper IV of the Capability Boundary Papers reports the case this addition is owed to.

9. Implications

For trans-capability ecosystem design. The empirical evidence supports Paper I's claim that the capability boundary is a design feature, not a limitation. The instrument's architecture makes the boundary explicit at every step: the AI system operates directively (analysis, comparison, assembly, verification), the architect

operates relationally (significance, deepening, architectural innovation), and a governed translation loop enables each to operate on material the other makes accessible. The resulting governance instrument exceeds what either participant could produce alone -- not by combining their capabilities additively but by making their interaction multiplicative through structured dialogue.

For the borrowed voice. The governance-level instance recorded at §2 confirms Paper I's account of the borrowed voice operating silently under capacity pressure. The AI system produced a fragment carrying the vocabulary of a governed revision. The failure was invisible to the system. It became visible only through the architect's intervention. The instrument's anti-masking verification -- a governed check requiring every substantive element to trace to a specific session source -- operationalizes Paper I's alerting mechanism at the governance level.

For the argument against replacement. The simulation data provides an attested progression consistent with the irreducibility claim. It is not a measurement a reader can re-run, and §3 says why; what it establishes is the shape of the distance, not its exact size. The directive system alone scores 3.1. The directive system with maximum analytical enhancement scores 4.4. The gap between 4.4 and 5.0 -- the distance the relational engagement closes -- is not a refinement gap. It is the architectural boundary. No scaling of the directive register produces what the relational register contributes. The architect's relational intelligence is not a supplement to the AI system's directive processing. It is a categorically different operation that produces categorically different results. The collaboration's value is irreducibly dyadic.

References

Heller, N. A. (2025). *Emergence of the Cube Problem: An Art-Based Research Inquiry.* Published manuscript.

The Capability Boundary Papers. Each states what it borrows at the point of borrowing; none is a prerequisite for another.

Heller, N. A. (2026). *The Capability Boundary Principle: Designing Intelligent Ecosystems Across Architectural Substrates.* Paper I of the Capability Boundary Papers.

Heller, N. A. (2026). *The Capability Boundary Principle Applied: A Diagnostic Map.* Paper II of the Capability Boundary Papers.

Heller, N. A. (2026). *The Borrowed Warrant: How a System's Own Encodings Acquire the Standing of Results.* Paper IV of the Capability Boundary Papers. Establishes the retention rule applied at §3 and the second failure at round two of the loop, treated at §8.

Heller, N. A. (2026). *The Capability Boundary Protocol.* Operational companion to the Capability Boundary Papers; converts them into gates.

Provenance. This is the generalized edition of a supporting note of the same title and date, written inside a particular research programme and reporting the development of one of its governance instruments.

The Capability Boundary Principle -- N. A. Heller -- April 2026 · Paper III of the Capability Boundary Papers