

OPEN
MAINFRAME
PROJECT

Modernization Working Group

Search for the Truth

A Reality Check on AI for Mainframe Modernization

Why the industry's evidence and its risk are measured at different scales

Executive Summary

In February 2026, a claim that AI could compress COBOL modernization "in quarters instead of years" [1] preceded a roughly 13% fall in IBM's share price in a single session its biggest single-day drop since October 2000 [2]. IBM answered the same day: "Decades of hardware-software integration cannot be replicated by moving code." [3], [4] Neither side published a measurement, a named customer, or a defect rate. Six months later, this research found no independent, published measurement at the scale at which either claim was made.

This paper reports what four industry practitioners and analysts told the Open Mainframe Project about where AI actually earns its place in mainframe application modernization, sets that testimony against the measured research that does exist, and asks a narrower question than the one the market keeps arguing about.

Its central finding is not that AI is overrated or underrated. It is that **the evidence and the risk are produced at different scales.** What is demonstrated and measured in this field translation of individual programs, explanation of individual files, benchmark pass rates is produced at the unit of the file. (Here a *unit* is the scope at which something is produced or judged; Section 3 defines the term.) What actually fails in modernization is produced at the unit of the system: dependencies, security models, workload scheduling, data flows, non-functional behavior, and the governance regime the application runs under. A claim proven at the first unit does not discharge risk carried at the second. That mismatch, not the underlying technology, is why the debate keeps failing to resolve.

From this follow the paper's practical conclusions:

- **AI is load-bearing where its output can be validated at the same unit at which it was produced.** That is why application understanding, documentation, dependency discovery and test generation are the strongest use cases in every source consulted and why autonomous conversion is not. The consensus finding becomes a rule that predicts where the next credible use case will appear.
- **The measured evidence is real, better than skeptics assume, and narrower than optimists assume.** Domain-adapted models reach 49.33% Pass@1 on COBOL generation benchmarks where most general-purpose open models score zero, and 34.93% on Java-to-COBOL translation, where general-purpose models score near zero; on COBOL-to-Java translation, by contrast, general-purpose models score as well or better [5]. The same researchers state that their work "does not capture cross-file dependencies, shared copybooks, or inter-program control flow that are common in real-world mainframe systems."
- **The economics cannot currently be settled, because the cost nobody measures is verification.** AI moves human effort from reading code to checking machine output. Net value is what is saved on the first minus what is added on the second. No source in this research measures the second term for mainframe modernization. Until someone does, return-on-investment claims in this market are unsupportable in either direction.
- **The strongest evidence-backed return available today is in automating verification rather than generation** the opposite of where the market's attention sits.
- **Human accountability is a governance requirement, not a temporary technical gap.** Regulatory and governance frameworks differ by application, industry and jurisdiction; they do not relax because a model improves.

The paper proposes one framework, explicitly as author synthesis: **evidence-gated modernization**, in which each stage of a modernization program is passed by evidence produced at the unit at which risk is carried, and no stage is passed because the output looks right.

It also states plainly what it cannot establish. Three of its four interviewees sell products or advisory services in this market and the fourth is an analyst whose firm advises vendors it covers. No enterprise end-user was interviewed. No survey data was collected. Nothing here was measured by the author.

1. Two Claims, Six Weeks Apart

On 23 February 2026, IBM shares fell about 13% reported as the company's biggest single-day drop since October 2000 [2] after Anthropic published a claim that its Claude Code tool could substantially compress COBOL modernization work. The claim was specific in ambition and unspecific in evidence: teams could "modernize their COBOL codebase in quarters instead of years," with AI automating the exploration and analysis phases that consume most modernization effort [1].

IBM responded the same day. Its senior vice president Rob Thomas argued that "translation captures almost none of the actual complexity" and that "decades of hardware-software integration cannot be replicated by moving code." [3]

Read the two positions carefully and the striking thing is not that they disagree. It is that **neither published a measurement.** No defect rates. No named production system. No comparison against what the previous generation of modernization tooling achieved on the same estate. Two assertions one of them followed by a fall of roughly 13% in a company's share price in a single session and no evidence on either side that a reviewer could check.

Six weeks later, Gartner published a Strategic Planning Assumption that more than 70% of mainframe *exit* projects initiated in 2026 would fail to produce their intended benefits, "due to an overestimation of generative AI tooling capabilities." [6] That figure has been widely repeated. It is a forecast about a specific class of project platform exits not a measured failure rate for modernization generally.

The same Gartner research names a mechanism worth pausing on: "aggressive investor demand for AI capabilities as the sole indicator of a vendor's long-term health forcing vendors to deploy AI even where unnecessary." [6] Two of the practitioners interviewed for this paper described the same phenomenon independently. Derek Britton: "I'm not aware of any mainframe software vendor incumbent or new entrant who isn't shouting about their latest AI facility." [7] Nicolas Newcomb, more bluntly: "Everybody is using AI… People tell you they're using AI. Just brush that to the side." [8]

Some evidence has appeared since, though not from independent parties: in February 2026 Swimm, a vendor of mainframe-code analysis tooling, tested Claude Code against its own platform on two Medicare COBOL programs [9]. Section 3 returns to the result.

When every vendor claims the capability, the claim stops carrying information. That is the environment this research was conducted in, and it is the reason the research was conducted at all.

2. What This Research Did

This paper is the product of the Open Mainframe Project Summer Mentorship 2026. Its first dispatch, published in July, argued that "will AI replace COBOL?" was the wrong question, and promised to answer a narrower one: where, specifically, in the chain from legacy code to modernized system, does AI create value that survives contact with production? [10]

Method

Four structured interviews were conducted between July and August 2026, each roughly thirty minutes (about two hours in total), following a common instrument covering current use, greatest value, business rule extraction, limitations, human expertise, validation practice, governance and advice. Each interviewee consented on record to being quoted. Later interviews carried forward claims from earlier ones for direct comment, which is how the disagreements in Section 4 surfaced.

The interview evidence was then set against: the Gartner research above [6]; a HyperFRAME Research brief [11]; the BMC Mainframe Surveys for 2025 and 2026 [12], [13]; peer-reviewed and preprint research measuring large language model performance on COBOL specifically [5] and on code translation generally [14]; randomized controlled trials of AI's effect on developers [15], [16]; a vendor's test of Claude Code on Medicare COBOL programs [9]; and the 2025 DORA (DevOps Research and Assessment) report [17], which is Google Cloud's research program and is unrelated to the EU's Digital Operational Resilience Act. Where a source made a claim this paper could not verify, the claim was left out.

How evidence is classified

The paper labels every substantive claim by the kind of evidence supporting it, because in this field those kinds are routinely conflated:

- **Measured research** - a benchmark, experiment or controlled trial with a published method.
- **Survey data** - with its sampling method stated, since sampling method is what determines what a survey can support.
- **Expert testimony** - an informed practitioner's observation, which is not a measurement.
- **Analyst research** - a commercial research product synthesizing client interactions.
- **Vendor claim** - a statement by a party selling the capability.
- **Illustrative example** - a rhetorical figure used to convey intuition, which is never a statistic.
- **Author synthesis** - a conclusion drawn by comparing sources, which no single source states.

The distinction that matters most is the last-but-one. Several arresting numbers circulating in this debate are illustrative examples that have hardened into statistics through repetition. This paper reproduces none of them as data.

Limitations, stated at the front rather than the back

The interviewee set is not neutral. Three of the four sell into this market: Gary Brach's company sells the grounding layer he argues is necessary; Nicolas Newcomb's sells deterministic transformation, which is why "GenAI is not the transformation engine" is a commercially convenient position for him; Derek Britton is a chief marketing officer at a vendor whose product is a disposable z/OS environment for exactly the kind of testing he identifies as the best use case. The fourth, Steven Dickens, is an analyst whose firm discloses that it "provides research, analysis, advising, and consulting to many high-tech companies, including those mentioned in" its own research [11].

There is no end-user in this set. The organizations that carry the risk are absent - from this corpus, and largely from public discussion, because failed programs are expensive to disclose. Every conclusion below should be read with that absence in mind.

Skepticism is also commercially motivated. This paper applies the same discount to cautious vendor claims that it applies to optimistic ones.

A survey instrument was written and circulated to the modernization working group; no responses are analyzed here. Nothing in this paper rests on survey data collected by the author.

Quotations are drawn from machine transcripts and have been corrected only for transcription artifacts, never for substance. Each quotation was checked against the transcript or written submission it comes from, and an ellipsis marks any passage omitted from within it.

This paper was researched and drafted with AI assistance. Anthropic, whose model was used, is a party to the February 2026 episode described in Section 1. All facts about that episode are taken from third-party press reporting and from the parties' own published statements [1], [2], [3], [4].

3. The Unit Problem

A note on terms. A *unit* is the scope at which something is produced, tested or judged. In increasing scale: a program; a job stream (a chain of batch jobs); an application; a system; and the organization and governance regime around it. The *unit of evidence* is the scope at which a given piece of evidence was produced, and the *unit of risk* is the scope at which a failure would be carried. The word is not used here in the sense of a *unit test*. Appendix C defines this and other terms.

Here is the observation this paper is built on.

Anthropic demonstrated capability by translating and analyzing **programs**. IBM rebutted at the level of **platform integration**. They were not, strictly, contradicting each other. They were describing different objects and reporting the results as if they were about the same one.

That pattern recurs everywhere in this field once you look for it.

In the measured research. The most rigorous COBOL-specific work available - the COBOL-Coder study, which fine-tunes models for COBOL and publishes results on purpose-built benchmarks - measures generation and translation at file level [5]. Its authors say so, and they say what that excludes: their approach "does not capture cross-file dependencies, shared copybooks, or inter-program control flow that are common in real-world mainframe systems," and their benchmarks lack "proprietary libraries, organization-specific coding standards, and operational constraints."

In practitioner testimony. Steven Dickens: "the mainframe is not a million lines of code running in a COBOL application. That COBOL application is running with CICS. It's also running with workload scheduling. It's running with RACF and ACF2 and Top Secret. There's a hundred things around it." [18] His estimate of the proportion -offered as illustration, not measurement - is that there are "400 questions and modernizing the COBOL is 10 of those questions."

In delivery experience. Derek Britton, on what a modernization program actually contains: "in most large-scale modernisation projects, there are 200, 300, 500 things that you will need to change, *even if the code stays the same*" - screen handling, where data resides, the disaster recovery plan, performance tolerances. "All of those things need to be defined in terms of: what do we expect good to look like, and then how do we measure that?" [19]

In scoping. Nicolas Newcomb: "we care about the boundaries - what's the system you want to get? So many times we find out that the customer's digging up more code as we go through the assessment." His assessment work then has to establish the database dependencies, the external dependencies, and "the things that need to be built out as functions that your system relies on that don't exist in a modern language" none of which is visible in any single program [8].

In time. Gary Brach's version is the same observation expressed as a sequence: "proof of concepts go really well, generally, in this space and then when you move to full implementation, they are disasters." [20] Scope expansion is what turns the demonstration into the failure.

And in the analyst research. Gartner notes that generative AI "does not account for the unique capabilities that the mainframe offers (e.g., ensuring that the same performance and throughput is achieved after the migration)." [6] HyperFRAME: "a superficial translation of COBOL into Java does not resolve these underlying architectural relationships." [11]

Six vantage points measured research, four practitioners and analysts approaching from different directions describing one structure:

> **Evidence in this field is produced at the unit of the file. Risk is carried at the unit of the system.**

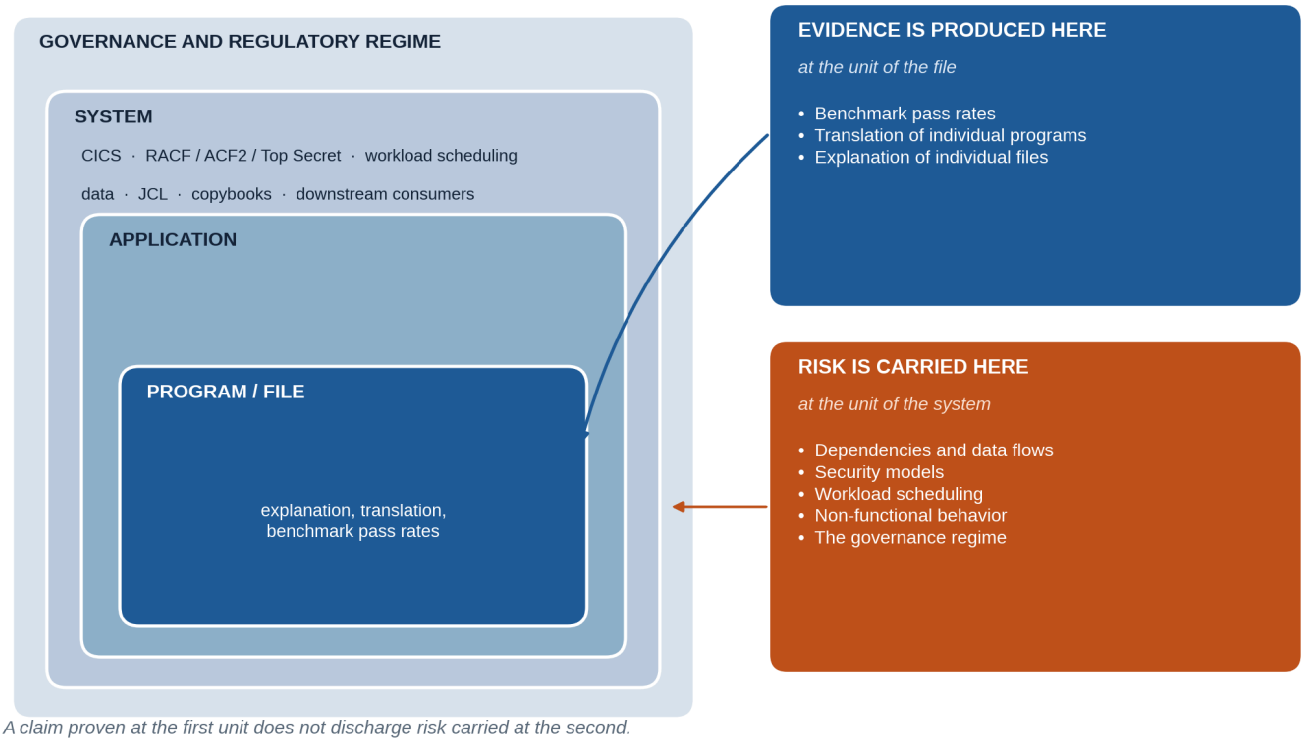


Figure 1 - The unit problem – evidence is produced at the unit of the file, while risk is carried at the unit of the system (author synthesis)

This is a synthesis. No source states it in this form. But it does explanatory work that the consensus position "AI is useful but needs humans" does not.

It explains the capability split rather than restating it. Every source in this research reports roughly the same list: AI is strong at understanding, explanation, documentation, discovery and test generation; weak at autonomous conversion and migration. That list is usually presented as an empirical accident of current model capability. It is not. Look at what the items have in common:

> **AI is load-bearing where its output can be validated at the same unit at which it was produced.**

An explanation of a program is checked against that program, by someone who can read it. A generated test is validated by running it. A dependency map is verified against the dependencies. In each case the unit of production and the unit of validation coincide, and a competent human can close the loop in bounded time.

There is a second condition, and it is easy to miss: the reviewer has to be able to detect what is missing as well as what is wrong. An error can be found by checking output against its source. An omission cannot, because nothing in the output points to it. Swimm's February 2026 test illustrates the risk [9]. Asked to extract the business rules of two Medicare COBOL programs of 4,692 and 18,835 lines, Claude Code mentioned 69–86% of the paragraphs of the smaller program across three runs, but only 24–35% of the larger one, and Swimm's audit of the extracted rules found some with conditions dropped. An explanation can be fluent, contain nothing false and still describe a third of the program, and a reviewer reading it against the code gets no signal that anything is absent. Swimm sells a competing product and tested one model on two programs, so this illustrates the problem rather than measuring its size. But it suggests that the rule needs a second clause: the validation loop closes only when a competent human can check completeness as well as correctness in bounded time. How large an output a reviewer can certify for completeness has not, as far as this research found, been measured.

One honest caveat on generated tests: running a test proves the test executes and the code behaves as the test expects. It does not prove the test checks the right thing. A test generated from existing behavior will faithfully encode existing bugs along with existing business rules which is exactly what you want when the goal is equivalence, and exactly what you do not want when the goal is improvement. The loop closes on correctness relative to current behavior, not on correctness in principle.

Autonomous conversion breaks the correspondence. The model operates on a program; correctness has to be established for a system. No amount of program-level confidence composes into system-level assurance, because the properties that fail at system level a security profile, a batch window, a throughput characteristic, a downstream consumer's assumption about a field, and mainframe specifics such as packed-decimal rounding, EBCDIC sort order, CICS transaction state or JCL restart behavior were never visible at the unit the model worked on.

It explains why demonstrations mislead without accusing anyone of dishonesty. February 2026 was not a lie. It was a true claim about one unit, received as a claim about another.

It gives buyers a question that discriminates. Not "does your tool use AI," which every vendor now answers yes. But: *at what unit is your evidence produced, and is that the unit at which I carry risk?*

And it is falsifiable. If a vendor publishes measured behavioral equivalence at system scope, under production-representative conditions, the argument concedes on that point. That is a specific, checkable condition, and this paper would rather be refuted by evidence than defended by hedging.

4. Four Routes to the Same Conclusion

The four people interviewed agree that human validation is required before production. It would be easy, and wrong, to report that as a consensus and move on. What makes the testimony useful is that they arrive there by four different routes, and each route supplies something the others do not.

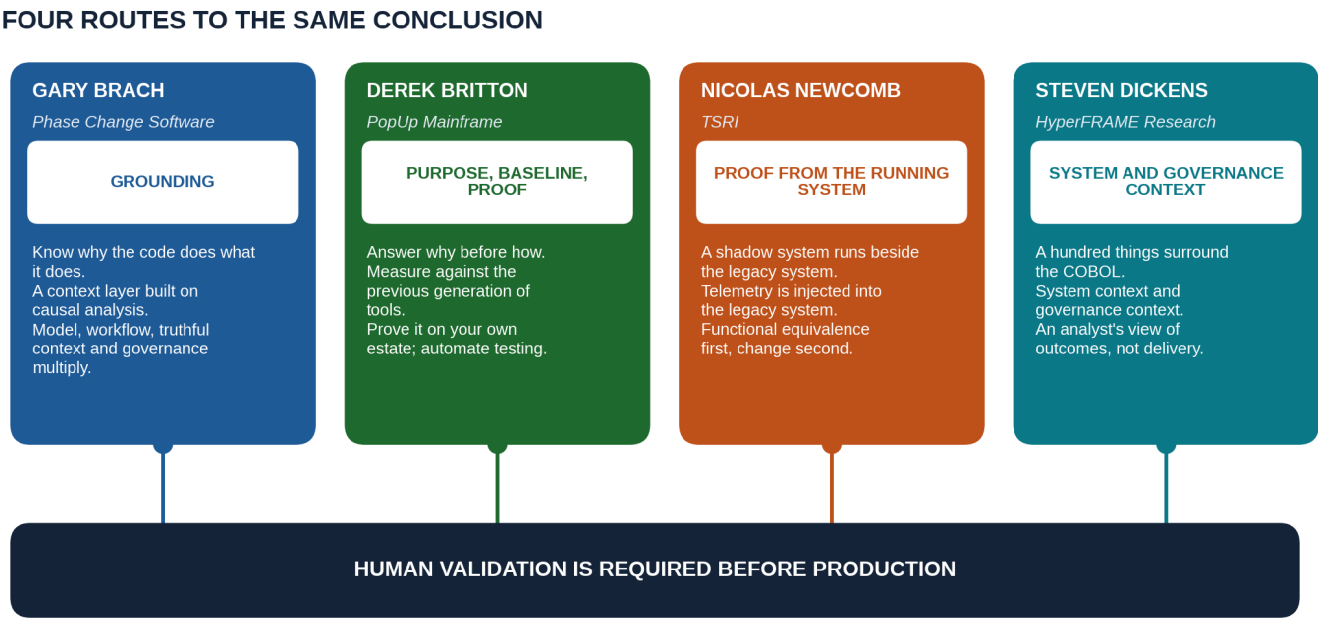


Figure 2 - Four routes to the same conclusion – how each interviewee arrives at human validation before production

Gary Brach — grounding

Brach, CEO of Phase Change Software, watches customers attempt AI-assisted modernization daily and describes three failure properties. For many of the use cases he sees, AI-assisted modernization is "not scalable" and "not trustworthy"; it is also "not traceable, and traceability is probably the biggest problem." [20]

His characterization of the failure mode is the most quotable sentence in this research:

> *"It sometimes works, sometimes it doesn't. Many times it doesn't. But it always sounds like it's working."*

His remedy is to ground the model in a layer of verified understanding an ontology, a context layer built on causal analysis rather than pattern matching. What matters, he argues, is not what the code does but *why*: “you want it to be able to link across, you know, concepts more than you want it to link across code.” He states the governing relationship crisply: “the more the LLM has to reason, the more it’s going to hallucinate. The less it has to reason, the better results you’re going to get.”

He is notably specific about where AI already works. Documentation is “certainly a great use case,” bounded by input quality. And on conversion he is more positive than his reputation would suggest: “it will actually migrate code really, really well because it’s a kind of language to language thing. *As long as you give it the right code to migrate*, it’s quite good at that… If you give it junk, it’s going to create junk.” What breaks down is “analysis at scale.”

On business rule extraction he pushes back on the industry’s framing. Rules alone are insufficient because they omit execution order: “you have to understand the computation plus the algorithm together… business rule extraction by itself is not enough. You’re going to miss the edge cases and you’re going to miss the flow, because you don’t understand the actual execution.”

He also offers the only structural model of adoption in this research, and it is a good one. Four factors the model, the workflow, the truthfulness of the context, and governance combine **multiplicatively**: “if those are wrong or zero, nothing’s going to work.” And then the test that should end most AI modernization business cases: “if all those things multiply and you’re getting the same thing that a human can do, then why are you doing it?”

Finally, a governance risk no one else raised. When an enterprise runs its codebase through a commercial model, “it is training on what you’re doing with it, and it is abstracting some of the things that you’re doing.” Brach’s claim is stronger than commercial terms support. Anthropic, whose model he used as his example, states that for its commercial products it does not by default use customers’ inputs or outputs to train its models [21]. The exposure is real for consumer accounts and wherever that default has been changed, and an organization should check its own contract rather than assume either way. The governance point survives the qualification: modernization exposes architecture and intent, not only source.

A note on one figure. Brach illustrated compounding error with an example about high per-transaction accuracy across many transactions. The intuition is sound and important: small per-unit error rates compose badly across large systems. The specific arithmetic in the example does not compute, and this paper does not reproduce it. It is an intuition, not a measurement.

Derek Britton — purpose, baseline, and proof on your own estate

Britton’s contribution is the most uncomfortable for everyone selling in this market, including his own employer [19].

He begins by refusing the premise that there is a right answer: “I’ve seen all of them tried, I’ve seen all of them succeed, and I’ve seen all of them fail. There is no one magic answer, and if there was, we would all be doing it. The right approach is dictated by the situation it’s not dictated by the technology.”

Then the observation this paper regards as the single most important methodological point in the corpus. Discovery and conversion tooling did not arrive with AI:

> *“By and large that is already quite a mature market in terms of discovery and conversion tooling. Good legacy modernization technology has existed for decades. So the delta isn’t very clear yet whether AI will truly change the game there, or by how much.” [7]*

In the interview he put it more sharply: “It’s really interesting that we think there’s a market that needs AI tools, but didn’t need non-AI tools to do that task… AI isn’t really the answer. AI has actually just created a debate that has caused people to want to answer those questions in a better way.”

Almost no claim in this market is measured against what the previous generation of tools already achieved on the same estate. Without that baseline, “AI improved discovery” is not a finding.

He identifies a constraint on capability growth that is commercial rather than technical. A model for COBOL understanding “will only be as good as the number of COBOL applications it’s been trained to look at” and “not many customers are happy to give up their COBOL applications.” The best data “is hidden behind the firewall of

these organizations that we're trying to serve." He calls this "an existential challenge that the AI world faces," and draws the market consequence: the winning vendors will need "good, loyal, and amenable customers" granting code access, not merely good technology. The COBOL-Coder researchers corroborate this from the other side, having had to build a bespoke data curation pipeline because suitable corpora did not exist [5].

On static versus dynamic understanding he is vivid: "you can't learn from the code alone what the system does… Should we look at a system by examining a painting of it, or should we look at the movie first, and just see the whole story?"

On validation he sets a bar that vendor references cannot clear: "no amount of shiny brochures saying look, we've done it at this other customer that doesn't prove anything to me until you've proven it on my banking system." What passes is a scoped pilot or phase-zero on a representative sample of the customer's own estate.

And in his written submission, the strongest return-on-investment argument in this research:

> *"…for every action (of changing something) you have to have the reaction of proving it is correct – which means, at least theoretically, testing [must] comprise half the effort of any modernization program. I think if all AI did was help automate the creation and execution of testing programs, it would pay for itself." [7]*

That estimate is his, and he hedges it himself. But the argument's structure holds regardless of whether the fraction is a half: the largest reliable cost in modernization is proving the change is correct, and that is where automation compounds. His published article makes the same point in softer terms: test design, test data creation, execution, comparison and defect analysis "consume a substantial proportion of a modernization program" [22].

Two further points from his written submission qualify the optimistic case in useful ways. First, the planning fundamentals are unchanged "cost, risk, skills, timescales, these remain the foundations of planning a modernization program." [7] Second, the cost model for AI itself is unsettled, and he draws the obvious analogy: "the backlash against cloud computing was about its hidden costs, the AI cost model is going to also have to play itself out." [7] Every ROI model in this space currently assumes an inference cost structure that has not stabilized.

He is also the only interviewee who credits AI with an effect that has already landed: "AI has forced everyone to rethink what's possible… users are daring to demand more, and vendors are scrambling to innovate." [7] That is a real change, with a real mechanism, and it is invisible to any code-level measurement.

Nicolas Newcomb — proof from the running system

Newcomb, CEO of TSRI, describes his firm as having been involved in hundreds of modernization projects through to fielded applications; that is TSRI's own account, not an independent count. His account differs from the market's in a way worth stating plainly: **he does not use generative AI to transform code.** [8]

> *"We don't bother using Gen AI to transform code because that would slow down our operations massively."*

His firm uses what he calls neurosemantic AI formal methods and symbolic representation to model systems and refactor them deterministically, then uses knowledge graphs to "inject accurate information about a system into large language models," with generative models applied to code-quality exploration and rule authoring. The automation that works at scale in his practice is not the generative kind.

His view of generative AI's industry-level effect is the sharpest counter-hype statement in this research:

> *"In some cases the use of Gen AI in modernization at an industry level has taken the industry back a decade… we're back in a scenario where everybody thinks they have a team of heroes… If you're using models that will just generate a whole bunch of stuff that you can't verify, you don't understand it, you aren't able to validate it and then you find out that, oh, it only did 70% of it."*

He also punctures the premise underneath most rewrite business cases. Customers arrive wanting to reimagine their workflows: "you're going to rewrite 90% of the business rules exactly as they existed before, and then maybe 10%. And that's something we rarely see either that somebody's prepared to change their business by 10%." His sequence is functional equivalence first, deliberate change second.

His answer to the validation problem is the most concrete in the corpus. TSRI builds a **shadow system running in parallel** with the legacy system and injects **telemetry into the legacy system** so both are heavily instrumented and their outputs can be compared through functional equivalence, performance and scalability testing, up to go-live. He is careful not to oversell it: this does not mean testing "can just be, push this out. It can't at all."

On barriers he is scathing about his own industry: "we exist in one of the least standardized industries, and a customer that is reaching into the industry of modernization is reaching into a grab bag of horses." He describes cleaning up after bait-and-switch engagements and "modernization groups that don't have engineers that have actually done modernization" a problem "made worse when folks are overconfident just because they have tools. Just because somebody gives you Photoshop doesn't make you a graphic designer."

And he identifies an organizational barrier that no amount of tooling addresses: the tenure of a typical CTO, which he puts at "usually two years." That figure is his own estimate; a Korn Ferry study of the 1,000 largest US companies, published in 2017, found an average CIO tenure of 4.3 years [23]. His argument survives the correction, because "the risk that is involved in picking up a large-scale modernization exceeds the tenure of that CTO" at either figure when the programs are long ones. Absent a clearly definable strategic purpose, programs become what he calls "zombie modernizations," dragging on five to nine years.

His buyer's heuristic is one that anyone about to sign a contract can use immediately:

> *"It's not about the AI… Everybody is using AI… People tell you they're using AI. Just brush that to the side. What have you done? How long did it take? Who did you do it for? What was the scale of that thing that you did?"*

Steven Dickens — the system around the code, and the governance around the system

Dickens is Principal Analyst and CEO at HyperFRAME Research, with fifteen years in the mainframe business at IBM and Broadcom before becoming an analyst. He is explicit about the limits of his vantage: "nothing specifically hands-on within an enterprise… I'm talking to a lot of the vendors and the clients that are actually doing that work." [18] He is an observer of outcomes, not a producer of them, and this paper cites him accordingly.

He places the value squarely on understanding: "so many of these shops have got a legacy code base, maybe the people who wrote the application have moved on. Having that foundational understanding and then the code explanation is kind of where the first value is." His estimate of the gain offered as an analyst's characterization, not a measurement is "days rather than years… a few people in days rather than hundreds of people in years," with the immediate qualifier that "it's not a one prompt and done type activity."

He is emphatic that tool choice is not incidental: "throwing Claude at a million lines of COBOL and expecting it to understand it just isn't going to happen… you're going to make some very expensive mistakes. And you're probably going to make more mistakes than if you did it manually." His prescription is tooling with mainframe context built in.

His central contribution is the nested-system argument quoted in Section 3. But the part of it that is most often dropped is the second axis. Dickens argues that there are *two* kinds of context, not one:

> *"You need the system context and you need the governance framework context to be able to have the conversation."*

A single bank may run two hundred mainframe applications under twenty different governance and regulatory frameworks. The same physical mainframe can host two applications with entirely different modernization requirements. "Trying to go in and say, oh yeah, we're going to modernize the bank all the same way no, it doesn't work that way."

This matters more than it first appears, because governance context is the one input AI cannot derive from the estate. Dependencies can be discovered from code and telemetry. Which regulator's framework applies to which application, and what that framework demands of a change, is external knowledge held by people.

Finally, Dickens volunteered something that materially affects how this paper and any paper should read the analyst literature:

> *"There's only really three or four analysts covering this space. We all think the same… We all know each other. We all share notes. We're all on the same page."*

He named two analysts he knows personally and shares notes with; one of the names he gave corresponds to a credited author of the Gartner research cited in this paper [6]. That does not make either firm wrong. It does mean that agreement between the major analyst voices in this debate should be counted **once, not twice**. Much of the apparent weight of the skeptical consensus rests on a professional community small enough to fit around one table.

Where they disagree

The disagreements are more informative than the consensus, and this paper does not smooth them over.

On grounding architectures, Britton declines to endorse Brach. Asked directly about the truth-layer thesis, he said: "I'm going to hold my hand up and say I don't know. I've evaluated, or rather have presented to me, a number of solutions in this area… they all develop variations on the same theme. But there wasn't a single one of them that I said, ah, well, that's obviously a better way than that one." Newcomb, asked a version of the same question, was unequivocal "large language models can't do anything without it." So: one vendor asserts grounding is the requirement, another vendor's practice confirms it, and an experienced evaluator who has seen several implementations cannot tell them apart. That is what an immature market looks like from the inside.

On generative AI's net effect so far, Britton and Newcomb describe the same year in opposite terms. Britton: AI has already helped, by raising ambition and forcing vendor innovation. Newcomb: it has set the industry back a decade by restoring false confidence. Both are experienced, both are describing 2026, and this paper does not adjudicate between them. It is possible likely, even that both are true of different organizations, which is precisely the DORA finding that AI amplifies whatever capability an organization already had [17].

On expert consensus itself. Dickens holds that everyone who knows the platform agrees, and that disagreement correlates with ignorance of it. That is a strong claim, and the experience of this research cuts against it: four informed people disagreed on grounding architecture, on generative AI's role in transformation, and on whether the last two years have helped or hurt. If Dickens were right, one interview would have sufficed.

5. What Is Actually Measured

Testimony is not measurement, and this paper's whole argument is that the distinction matters. So what has actually been measured?

COBOL specifically

The COBOL-Coder research provides the only COBOL-specific measured results found in this study [5]. It fine-tunes a code model on curated COBOL and evaluates on purpose-built benchmarks COBOLEval (146 tasks) [24], COBOLCodeBench (46 harder tasks), and a new bidirectional COBOL–Java translation benchmark (143 pairs).

The results deserve to be read carefully by both camps.

For the skeptics: general-purpose models are poor at COBOL. On COBOLEval, most open-source baselines score 0% Pass@1; GPT-4o reaches 16.4%, with a compilation success rate of 41.8%. This is measured support for Dickens's warning against generic tooling.

For the optimists: domain adaptation works where the target is COBOL, and works quickly. The 14B domain-adapted model reaches 49.33% Pass@1 and 73.95% compilation success on COBOL generation. On COBOL-to-Java translation it reaches 83.91% Pass@1, but general-purpose models do as well or better there

(GPT-4o 85.31%, GPT-oss-120B 89.51%), so this result shows that translating out of COBOL is tractable, not that domain adaptation is needed for it. Anyone claiming AI cannot translate COBOL is arguing against measured results.

For everyone: the direction matters enormously. Java-to-COBOL translation reaches only 34.93%, with baseline models scoring near zero. Generating into COBOL is far harder than generating out of it a finding with direct implications for anyone planning to modernize *on* the platform rather than off it.

One qualifier applies to all of these figures and is easy to lose. COBOLEval is adapted from HumanEval [25], a benchmark of small, self-contained programming problems. “Pass@1” means the model’s first attempt satisfies the task’s test cases. These are individual, isolated programs not a program that reads a copybook, calls three other modules, checks SQLCODE, and has to restart cleanly halfway through a batch window. That is not a criticism of the research, which is careful about its own scope. It is the point: this is the best measurement available, and it is measuring something considerably smaller than the thing enterprises are deciding about.

And the limitation, in the authors’ own words, is the argument of this paper: the method treats “each source file as an independent unit” and “does not capture cross-file dependencies, shared copybooks, or inter-program control flow that are common in real-world mainframe systems.” Their practitioner reviewers three of them, a limitation the authors state said production adoption would require predictable handling of “file status checks, handling of SQLCODE… and restartability,” adherence to organizational copybooks and standards, and understanding of JCL and cross-program dependencies. The authors conclude that LLMs “are most effective as human-in-the-loop draft generators,” and their practitioners compared the best usage pattern to a junior developer producing an initial draft.

That description is the most defensible single characterization of current capability this research found, and it came from a measurement rather than an opinion.

The measurement layer is immature in both directions

A paper that cited only the low benchmark numbers would be cherry-picking. In May 2026, researchers at Concordia examined why LLM code translations fail, manually inspecting 150 failures sampled from roughly 600 observed across 6,164 translations [14]. A substantial share were **evaluation-induced** missing library linking, wrong compilation flags, faulty test conversion with a 5–10% error rate rather than genuine semantic failures.

In other words: published translation accuracy figures may *understate* model capability. Both the optimistic and the pessimistic readings of the benchmark literature are less secure than they appear, which is itself a finding about the maturity of this evidence base.

What happens when humans use these tools

The best-known randomized controlled trial of experienced developers is METR’s 2025 study of open-source developers [15]; it is not the only one. Sixteen developers completed 246 real issues in repositories they had contributed to for years. The result:

> Developers took **19% longer** to complete issues when allowed to use AI tools while estimating afterwards that AI had made them **20% faster**.

This is not a mainframe study, the sample is small, and the authors explicitly decline to generalize it. It should not be used to claim AI slows down modernization work. What it does establish, in a setting where the question was randomized, is that **self-reported productivity gains from AI are not reliable evidence of productivity gains.** That is Gary Brach’s “it always sounds like it’s working,” measured.

The 2025 DORA report points the same way at industry scale: 90% of technology professionals use AI at work and more than 80% believe it has increased their productivity, while 30% report little or no trust in AI-generated code and higher AI adoption correlates with increases in **both** delivery throughput **and** delivery instability [17]. DORA’s framing is that AI is an amplifier: “it magnifies the strengths of high-performing organizations and the dysfunctions of struggling ones.”

That is a more precise version of what every interviewee said about human expertise. AI does not supply engineering discipline. It multiplies whatever discipline is already there in either direction.

The METR result also has company. Three field experiments at Microsoft, Accenture and a Fortune 100 company, covering 4,867 developers, found that those given an AI coding assistant completed 26.08% more tasks [16], and METR's own February 2026 update says its newer data suggests the picture has moved since 2025 but is unreliable because of selection effects [15]. The measured effect evidently varies with the tool, the task and the developer, and none of it was measured on mainframe work.

Adoption, and the premise underneath it

BMC's 20th annual Mainframe Survey, fielded 31 March to 15 April 2025 with more than 1,000 respondents, reports that 65% are already using generative AI with the mainframe and 74% consider it critical to their mainframe strategy within two years [12]. The survey is vendor-run and email-invited, and its respondents are self-selected; it establishes direction of travel, not proportions of a population, and no margin of error can be claimed for it. BMC's 2026 survey, published on 1 September 2026 with more than 1,300 respondents, found that for code management 40% of respondents are willing to have AI recommend actions but only 23% are willing to let it complete them [13]. That pattern fits the interviewees' insistence on human validation. This paper has not examined the 2026 survey's sampling method, so the caution above should be assumed to apply to it too.

One figure in it deserves more attention than it has received. **66% of respondents identify as millennial or Gen Z, up from 37% in 2018.**

The retiring-workforce premise is one of the two load-bearing justifications for modernization urgency, and this is the best available data on it pointing the other way. HyperFRAME makes the same argument differently: the skills gap "is often framed as a people problem, but it is really a knowledge representation problem" the knowledge exists, it is simply not written down in a form modern teams can consume [11]. Britton contests the second premise as directly: the difficulty of COBOL is "based on an unproven assumption," and COBOL in 2026 "is perfectly fine as an enterprise application option, has itself modernized as a language, and is fully supported across all modern environments." [7]

Neither premise is refuted. Both are contested inside the evidence base, and both are routinely used to justify eight- and nine-figure programs.

6. Where AI Is Load-Bearing Today

The table below is this paper's assessment. It is **author synthesis**, not an industry benchmark, and it is not statistically validated. What is new is not the ranking but the third column, which explains the ranking rather than asserting it.

Activity	Assessment	Why (unit of validation)	Strongest evidence
Code explanation and application understanding	Strongest current use case	Output is checked against the same program it describes	Unanimous across four interviews; Gartner [6]; HyperFRAME [11]
Documentation generation	Strong, bounded by input quality	Same unit; a human who knows the system can judge it	Brach; Dickens; HyperFRAME
Dependency and impact discovery	Strong within scope, incomplete at system scope	Discovered dependencies are verifiable; undiscovered ones are invisible	Gartner (GenAI is "transforming the 'discovery' phase"); Newcomb
Test and test-data generation	Strong, and under-exploited	Tests are validated by running them	Britton (written [7]); HyperFRAME; Gartner
Operational support and diagnostics	Growing, distinct from modernization	Suggestions are checked against live behavior	Gartner (CICS and JES agents); BMC surveys [12], [13]; HyperFRAME
Business rule extraction	Useful but structurally incomplete	Rules are file-scoped; behavior is flow-scoped	Brach [20]; contested framing

Activity	Assessment	Why (unit of validation)	Strongest evidence
Bounded code transformation	Real, measured, input-dependent	Correctness measured at file level; risk sits above it	COBOL-Coder [5] (measured); Brach
System-scope autonomous conversion	Not supported by available evidence	Correctness can only be established above the unit of operation	Gartner; Newcomb [8]; Dickens [18]; COBOL-Coder's own limitations
Proof of behavioral equivalence	Not an AI capability	The evidence must come from the running system	Newcomb; HyperFRAME on behavioral replay
Governance and approval	Not delegable	Accountability is external to the estate	Dickens; all interviewees

Read down the "why" column and the pattern is the whole argument: the activities at the top are the ones where a competent human can close the validation loop at the same unit at which the machine worked. The activities at the bottom are the ones where they cannot.

A note on business rule extraction. It appears in the middle of this table rather than the top because Brach's objection is structural rather than a complaint about quality: "business rule extraction by itself is not enough. You're going to miss the edge cases and you're going to miss the flow, because you don't understand the actual execution." A rule is a file-scoped object; the behavior that matters is a flow-scoped one. The same unit mismatch appears one level down.

7. The Verification Bottleneck

This section makes an argument no source in this research makes, though several supply its parts.

Every source agrees that AI shifts human effort. HyperFRAME states it most directly: AI "does not eliminate the need for experienced mainframe talent, but it changes where that talent is most valuable: from manually decoding every line of logic to validating, governing, and prioritizing AI-assisted recommendations." [11]

Set that beside Brach's description of practice the model produces a step-by-step account of what it did, and a human audits it "just like a manager checking the work" and beside his multiplicative test: "if you're getting the same thing that a human can do, then why are you doing it?" [20]

The economics follow immediately:

> **Net value = (comprehension effort avoided) − (verification effort added)**

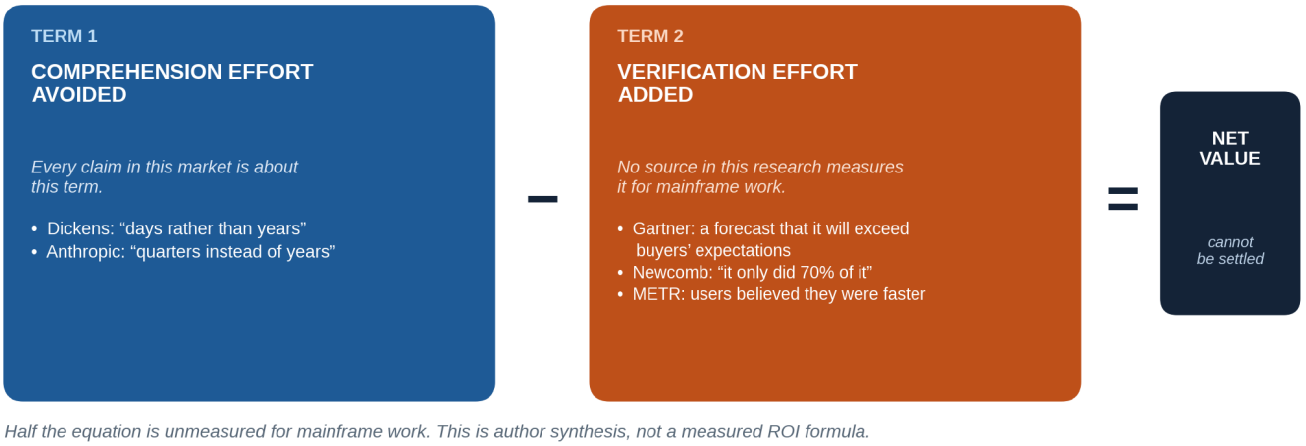


Figure 3 - The verification bottleneck – net value is comprehension effort avoided minus verification effort added (author synthesis, not a measured ROI formula)

Every claim in this market is about the first term. **No source in this research measures the second for mainframe modernization.**

Adjacent evidence exists for software development generally. A 2026 review of AI-assisted coding reports telemetry from Faros AI, covering 22,000 developers across 4,000 teams, in which review time rose 441% against a 33.7% rise in tasks completed [26], and DORA's finding that higher AI adoption correlates with greater delivery instability is at least consistent with it [17]. The telemetry is vendor-collected and is cited here as reported in the review, and neither source is specific to mainframes. Neither settles the cost question, but both show that the second term can be measured.

That is why the return-on-investment question cannot currently be settled. It is not that the answer is unfavorable; it is that half the equation is unmeasured. Dickens's "days rather than years" is a claim about term one [18]. Anthropic's "quarters instead of years" is a claim about term one [1]. Gartner's forecast of exit-project failure is a prediction that term two will be larger than buyers expect [6]. Newcomb's "you find out that, oh, it only did 70% of it" is a description of term two arriving late and all at once [8]. METR's result is what it looks like when term two quietly exceeds term one and nobody notices, because the participants still believed they were faster [15].

Three practical consequences follow.

First, ROI claims in this market are currently unsupportable in either direction. This paper does not assert that AI-assisted modernization fails to pay for itself. It asserts that nobody has published the mainframe-specific measurement that would settle it, and that a buyer should treat any confident figure optimistic or pessimistic as an estimate rather than a finding.

Second, verification effort should be instrumented from day one. Any organization running an AI-assisted modernization pilot can measure this cheaply: time-to-understanding with and without assistance, and review hours per unit of AI output, tracked separately. Reporting both terms would do more for this field than another capability assessment. It is also the single most valuable contribution an enterprise could make to the public evidence base.

Third and this is the recommendation this paper would most like practitioners to take away the highest-confidence return available today is in automating verification, not generation.

That inference is Britton's (Section 4): testing is, on his estimate, about half the effort of a modernization program, so automating it would pay for itself [7]. Notice why this argument is stronger than the generation case even though the estimate behind it is softer. Generated tests are validated by running them the unit of production and the unit of validation coincide, so the verification bottleneck does not reappear one level down. Automating verification is the one intervention that attacks the term everyone else is ignoring.

The market's attention is almost entirely on the other term.

8. Evidence-Gated Modernization

The framework below is **author synthesis**. It is not an industry standard, it has not been validated against outcome data, and it should be treated as a structure for argument rather than a certification scheme.

Its inputs are the four routes to trust described in Section 4. Its contribution is not the sequence variants of which appear in many modernization methodologies but the two right-hand columns, which specify what passes each stage and at what scope the evidence must be produced.

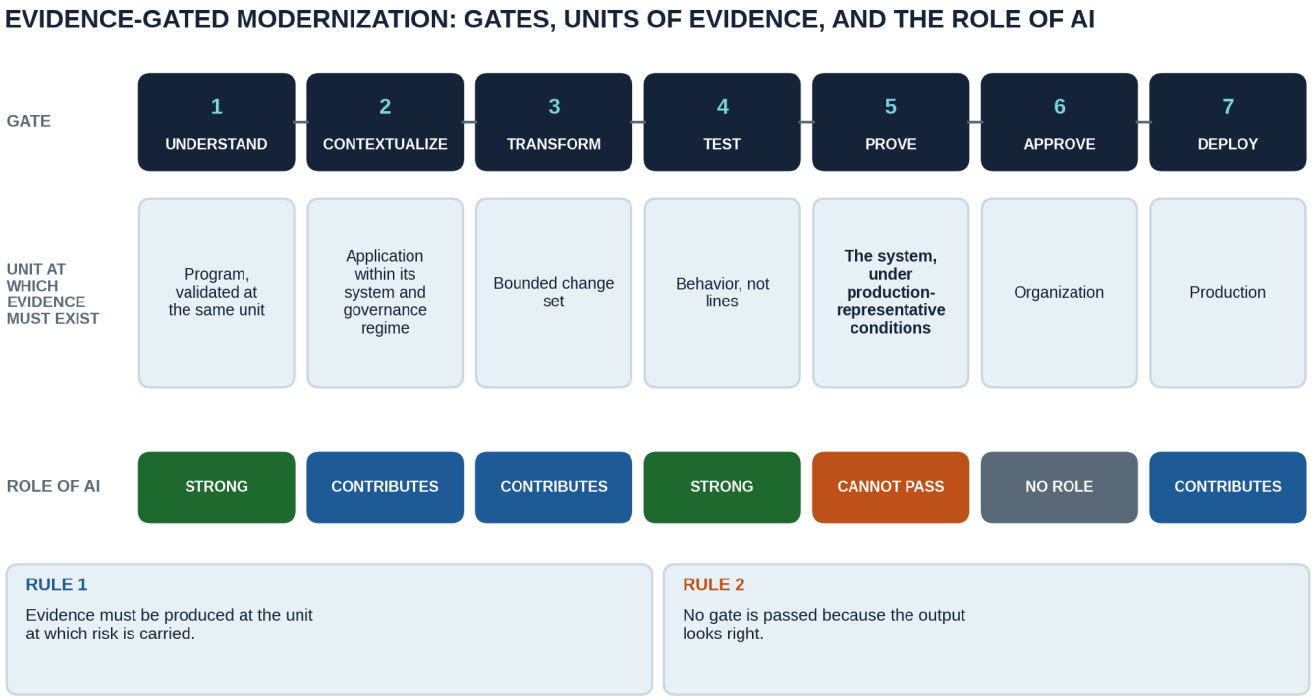


Figure 4 - Evidence-gated modernization – the seven gates, the unit at which evidence must exist, and the current role of AI (author synthesis)

Gate	The question	What passes it	Unit at which evidence must exist
1. Understand	What does this code appear to do?	Explanation, dependency map and extracted rules, each traceable to source and reviewable by someone competent to judge them	Program validated at the same unit
2. Contextualize	What is the system, and what governs it?	An agreed and frozen system boundary; enumerated data, security, scheduling and integration dependencies; the applicable governance and regulatory frameworks named	Application within its system, and its governance regime
3. Transform	What changed, and why?	Traceable mapping from source to target, and a recorded business purpose for the change	Bounded change set
4. Test	Do the tests exercise what matters?	Coverage of the behaviors identified at gates 1–2, including exception paths and edge cases	Behavior, not lines
5. Prove	Does the changed system behave as required?	Comparison against observed behavior of the running system, plus non-functional equivalence: performance, throughput, recovery, security posture	**The system, under production-representative conditions**
6. Approve	Who is accountable?	A recorded decision by a named human competent to make it under the applicable framework	Organization

Gate	The question	What passes it	Unit at which evidence must exist
7. Deploy	What happens if we are wrong?	Controlled rollout, monitoring against the gate-5 baseline, tested rollback	Production

Two rules make this a decision procedure rather than a checklist.

Rule 1 — Evidence must be produced at the unit at which risk is carried. A gate is passed by evidence at its own unit. Evidence from a smaller unit does not pass a larger gate. A correctly translated program does not pass gate 5; only the system does. This is the operational form of Section 3, and it is what makes the framework falsifiable rather than merely cautious.

Rule 2 — No gate is passed because the output looks right. Plausibility is not evidence. This rule exists to block exactly the failure Brach describes and METR measured.

Proportionality. This framework does not demand a shadow system for every estate. Gate 5 requires evidence proportionate to the consequence of being wrong: for a mission-critical transactional system that means production-representative behavioral comparison of the kind Newcomb's practice and Mechanical Orchard's behavioral replay approach both implement; for a peripheral reporting job a well-designed regression suite may suffice. What is never proportionate is passing gate 5 with gate-1 evidence which is precisely what a demonstration-led narrative invites.

Prior art. Behavioral equivalence as the success criterion for modernization is not this paper's idea. HyperFRAME describes Mechanical Orchard's approach capturing production inputs and outputs so that "behavioral equivalence becomes the primary metric of success" [11] and Newcomb's firm arrives at the same requirement independently through shadow systems and telemetry injection. This framework's contribution is to say *where in a program that bar applies*, and what it means for the gates before it.

Where AI sits in this structure is now a consequence rather than an assertion: it is strongest at gates 1 and 4 the two gates whose evidence is produced and validated at the same unit. It contributes at gates 2, 3 and 7. It cannot pass gate 5, and it has no role at gate 6.

9. Governance, Vendors, and the Shape of the Market

Three findings sit outside the technical argument but bear directly on outcomes.

Governance context is a second axis, and AI cannot derive it. Dickens's observation that a bank may operate twenty governance frameworks across two hundred applications means modernization requirements are not a property of the code. They are a property of the code *plus* the regulatory regime it serves. Dependencies can be discovered from source and telemetry. Which framework applies, and what it demands of a change, is external knowledge held by people. This is the clearest reason why human accountability at gate 6 is structural rather than transitional: it is not waiting on a better model.

Vendor durability is a modernization risk. Alongside its widely quoted exit-project forecast, Gartner published a second Strategic Planning Assumption: by 2030, 75% of vendors operating in the mainframe exit market will either pivot their business models or cease to exist [6]. Set that against Newcomb's description of "one of the least standardized industries" and "a grab bag of horses," and vendor selection becomes a survivability question as much as a capability one. A modernization program outlasting its tooling vendor is a specific, foreseeable failure mode.

Two well-placed sources describe AI capability claims as non-differentiating. Gartner names investor demand as forcing vendors to deploy AI "even where unnecessary"; Britton observes that no vendor is not shouting about AI; Newcomb advises buyers to set it aside. The practical consequence is that the AI question in a procurement is close to worthless, and the questions that discriminate are the older ones what have you done, for whom, and at what scale (Section 4) and Britton's condition that none of it counts "until you've proven it on my banking system."

10. What This Research Does Not Establish

Stated plainly, because a paper about evidence standards that exempted itself would not deserve to be read.

This research measured nothing. It conducted four interviews and read the available literature. Every quantitative figure in it belongs to someone else, and is labeled with its source and its kind.

It cannot support any universal claim about return on investment, productivity, accuracy, cost reduction or project duration for AI-assisted mainframe modernization. Nor can it support the negative versions of those claims.

No enterprise end-user was interviewed. Three interviewees sell into this market; the fourth is an analyst whose firm advises vendors it covers. Their observations are informed and, on several points, commercially inconvenient to themselves but the people who carry the risk are not represented here.

Analyst agreement in this field should be counted once. By the account of one of its members, the analyst community covering this space is three or four people who share notes and are "all on the same page." That is not a criticism of their work. It is a statement about how much independent confirmation it represents.

No survey data was collected. A survey instrument was written and circulated; no responses are analyzed in this paper.

Benchmark results have a short shelf life. The COBOL-specific figures cited here reflect a particular set of models at a particular moment, and the research on false failures suggests the measurement methodology itself is still settling. Britton's warning applies to this paper as much as to any: "your challenge with a white paper is to say, at this point in time, it's this." [19]

The evidence is stronger for identifying risks and promising use cases than for quantifying benefits. That asymmetry is the honest state of this field in 2026, and no amount of confident writing changes it.

It covers modernization on the platform only thinly. Most of the evidence reviewed concerns translation and exit. Modernization in place refactoring COBOL, running Java on z/OS, API enablement, DevOps on IBM Z, and AI for operations enters only through Gartner's observation that for many large organizations AI is more an enabler of in-place modernization than of migration [6], and through the operational-support row of Section 6. It deserves its own evidence review.

What would change these conclusions

- Published measurement of AI-assisted modernization at **system scope** one production application, reporting defect escape, rework, validation coverage, non-functional equivalence and elapsed time, including the failures.
- Any published measurement of **verification effort** alongside comprehension effort on mainframe work.
- A **counterfactual** comparison against the previous generation of discovery and conversion tooling on the same estate.
- **End-user testimony**, ideally anonymized and collected through a neutral body such as the Open Mainframe Project.
- A community-maintained list of **what must be equivalent after modernization** functional, operational, non-functional, security and compliance. Britton counts 200 to 500 such items; Dickens counts 400 questions. Nobody has published the list. Of everything identified in this research, this is the contribution the mainframe community could most usefully make next.

To make that last item concrete, here is an illustrative starting sketch of what such a list might cover. It comes from peer-review input by Nithin Sai Jalukuru rather than from this paper's evidence, it has not been validated, and it is offered only to show how much of what must be equivalent lies outside any single program:

- **Data:** EBCDIC sort order; COMP-3 packed-decimal precision and rounding; LOW-VALUES and HIGH-VALUES; REDEFINES.
- **Files:** VSAM file types and indexes; GDG generations; DFSORT sort order.
- **Transactions:** CICS transaction state; SYNCPOINT and rollback; Db2 locking.
- **Batch:** JCL condition codes; checkpoint and restart; scheduler dependencies and triggers.
- **Security and audit:** RACF, ACF2 or Top Secret profiles; SMF audit records.
- **Operations:** disaster recovery; GDPS failover behavior; service level agreements.

11. Recommendations

A word on how to read these. "Not proven" is not the same as "do not do it." BMC's 2025 survey suggests roughly two-thirds of mainframe organizations are already using generative AI on the platform [12]; the adoption decision has largely been taken. These recommendations assume you are doing this, and are about doing it in a way that produces evidence rather than confidence.

For organizations planning modernization

1. Answer *why* before *how*. Every interviewee said some version of this; Britton and Newcomb said it most forcefully. A program without a definable business purpose becomes a zombie program regardless of tooling.
2. Do not confuse migration with modernization. Gartner notes vendors routinely conflate them, and that for many large organizations AI is "more of an enabler for modernization in-place than migration off the mainframe." [6]
3. Define what "good" looks like, and how it will be measured, before starting including the non-functional tail that survives even when the code does not change.
4. Ask vendors the questions that discriminate: what have you done, how long did it take, for whom, at what scale and require proof on a representative sample of *your* estate.
5. Ask at what unit their evidence is produced, and whether that is the unit at which you carry risk.
6. Instrument verification effort from the first pilot. It is the number the whole industry is missing, and you can produce it cheaply.
7. Start with understanding and testing. That is where the evidence supports value today, and where the validation loop closes cleanly.
8. Assume vendor turnover. Gartner forecasts three-quarters of exit-market vendors pivoting or disappearing by 2030; choose accordingly.

For vendors

9. Publish evidence at system scope. The field's credibility problem is not skepticism; it is that the most consequential claims are made at the unit that is easiest to demonstrate.
10. Report the counterfactual. What did the previous generation of tooling achieve on the same estate?
11. Report verification effort, not only generation speed.

For the Open Mainframe Project community

12. Sponsor the equivalence checklist described above.

13. Create a route for anonymized enterprise outcome reporting, including failures. The absence of end-user evidence is the largest single gap in this field, and it exists because disclosure is costly which is a coordination problem an open foundation is unusually well placed to solve.

12. Conclusion

The question this research set out to answer was not whether AI will transform the mainframe. It was where AI can produce useful modernization work, what evidence makes that work trustworthy, and where humans must remain accountable.

The answer is more specific than the consensus and, this paper argues, more useful.

AI is genuinely load-bearing today in application understanding, documentation, dependency discovery, test generation and operational support. It is not load-bearing for autonomous transformation of production systems. That split is not an accident of current model capability and it will not be resolved simply by better models, because it follows from something structural: **AI earns trust where its output can be validated at the same unit at which it was produced, and modernization risk is carried at a larger unit than the one these models operate on.**

Everything else in this paper follows from that. It is why demonstrations at program scope keep failing to predict outcomes at system scope. It is why the February 2026 exchange produced a market movement instead of a conclusion. It is why the practitioners who have actually fielded these systems all insist on evidence from the running system, and why the researchers who measured the models said the same thing in their limitations section. And it is why the most valuable thing an organization can do with AI right now may be to point it at verification — the one place where automating the work does not create more work to check.

Gary Brach's sentence is the one to keep:

> *"It sometimes works, sometimes it doesn't. Many times it doesn't. But it always sounds like it's working."*

The task is not to decide whether AI is good or bad for mainframe modernization. It is to build the habit of asking, every time: *at what scale was this proven, and is it the scale at which I will be wrong?*

Appendix A — Contributors and Attribution

Gary Brach, Chief Executive Officer, Phase Change Software. Interviewed 17 July 2026. Quoted with permission [20].

Derek Britton, Tech Marketing Leader and Advisor, and CMO at PopUp Mainframe. Interviewed 20 July 2026 [19]. Also cited from a written submission provided to this research in July 2026 [7], and from his published article of 29 July 2026 [22]. Quoted with permission; the written submission was provided for use in this white paper.

Nicolas Newcomb, Chief Executive Officer, TSRI. Interviewed 25 July 2026. Quoted with permission [8].

Steven Dickens, Chief Executive Officer and Principal Analyst, HyperFRAME Research. Interviewed August 2026. Quoted with permission [18]. Cited throughout as an analyst rather than a practitioner, per his own description of his vantage.

With thanks to **Misty Decker** for mentorship throughout this project, and to the Open Mainframe Project and the Linux Foundation for the Summer Mentorship 2026 program. Misty Decker's guidance shaped this paper's method and evidence discipline; she is not cited as a research source.

With thanks to **Nithin Sai Jalukuru**, an application programmer working on mainframe systems, for a detailed peer review of this paper. His comments led to corrections to several figures and sources, a clearer definition of "unit of evidence," and the starter list of mainframe equivalence items in Section 10.

Appendix B — Evidence Register

Source	Type	Weight given	Principal caveat
Four industry interviews [8], [18], [19], [20]	Expert testimony	High for observation, none for measurement	Three vendors, one analyst; no end-user
Britton written submission [7] and article [22]	Expert testimony, prepared	High	Vendor perspective, so described by the author
Gartner, *Too Big to Fail* (8 Apr 2026) [6]	Analyst research	High for framing	Forecasts are not measurements; stated basis is "over 2,500 client interactions," i.e. practitioner synthesis, not controlled study
HyperFRAME, *Mainframe Modernization in the AI Era* (Jul 2026) [11]	Analyst research	Moderate–high	Co-authored; firm discloses advisory relationships with companies discussed; overlaps the Gartner analyst community
COBOL-Coder (arXiv) [5]	Measured benchmark	Highest in this paper	File-level scope; three-practitioner qualitative panel; model-generation sensitive
Beyond Translation Accuracy (arXiv, May 2026) [14]	Measured	Moderate	Not COBOL; shows accuracy figures can understate capability
METR RCT (Jul 2025) [15]	Randomized trial	Moderate, tightly scoped	Not mainframe; n=16; authors decline to generalize; its February 2026 update calls its newer data unreliable
Cui et al., three field experiments (Management Science, 2026) [16]	Randomized field experiments	Moderate	Not mainframe; measures completed tasks, not correctness or verification effort
DORA 2025 [17]	Industry survey	Moderate	Correlational; not mainframe-specific
BMC 20th Annual Mainframe Survey [12]	Vendor survey	Moderate for direction	Self-selected, email-invited; no margin of error claimable
BMC 2026 Mainframe Survey [13]	Vendor survey	Moderate for direction	Sampling method not examined by this paper
Swimm test of Claude Code (Feb 2026) [9]	Vendor test	Low–moderate; illustrative	Vendor with a competing product; one model, two programs, three runs each
Review of AI-assisted coding, citing Faros AI telemetry (Aug 2026) [26]	Secondary report of vendor telemetry	Low–moderate	Cited as reported; not mainframe-specific
Feb 2026 Anthropic/IBM episode [1], [2], [3], [4]	Press record and published statements	High for the fact of the exchange	Neither party published supporting evidence

Numbers deliberately excluded from this paper: any figure for the cost of a failed modernization program; any per-transaction accuracy arithmetic; any vendor accuracy claim expressed as a decimal; any availability figure not corroborated across sources; any survey result the author did not verify at source.

Appendix C — Glossary

Behavioral equivalence. The state in which a modernized system produces the same outputs and side effects as the system it replaces for the same inputs, and matches its non-functional behavior (performance, throughput, recovery, security posture), under production-representative conditions.

Compilation success rate. The share of model-generated programs that compile without error. It says nothing about whether they behave correctly.

Copybook. A COBOL source file of shared data-structure definitions that many programs include. A change to a copybook can affect every program that uses it.

GDG (generation data group). A z/OS facility that keeps related datasets as numbered generations, so that jobs can refer to the current or a previous version.

Parallel run. Operating the legacy and modernized systems side by side on the same inputs and comparing their outputs. A shadow system is a parallel run in which the new system's results are not yet used in production.

Pass@1. The share of benchmark tasks for which a model's first attempt passes the task's test cases.

Strategic Planning Assumption (SPA). Gartner's term for a dated forecast about a defined outcome. It is an analyst's judgment, not a measurement.

Unit of evidence. The scope at which a piece of evidence was produced, for example a program, an application or a system. See Section 3.

References

[1] Anthropic: *How AI helps break the cost barrier to COBOL modernization*. Claude blog, 23 February 2026. Link

[2] Ghoshal, A.: *Anthropic's claim that AI can quickly refactor COBOL rattles IBM investors*. CIO, 25 February 2026. Link

[3] Thomas, R. (IBM): *Lost in Translation: What the AI code debate keeps getting wrong*. IBM Newsroom, 23 February 2026. Link

[4] Kobie, N.: *Anthropic says Claude Code can help streamline 'cost-prohibitive' COBOL modernization, but IBM says it's not that simple*. IT Pro, 24 February 2026. Link

[5] Dau, A. T. V.; Tan, S. H.; Yang, J.; Bui, N. D. Q.; Nguyen, A. T.: *COBOL-Coder: Domain-Adapted Large Language Models for COBOL Code Generation and Translation*. arXiv:2604.03986, April 2026. Link

[6] Smith, D.; Galimberti, A.; et al. (Gartner): *Too Big to Fail: Why Mainframe Exit Projects Are Likely to Fail in the Age of Generative AI*. 8 April 2026, ID G00850207. Licensed client reprint. Link Public press release: Gartner Predicts More Than 70% of Mainframe Exit Projects Will Fail Due to Overestimation of Generative AI's Capabilities, 18 June 2026. Link

[7] Britton, D.: *Additional Input on Mainframe Modernization and AI*. PopUp Mainframe, July 2026. Written submission provided to this research; cited with the author's permission for the Open Mainframe Project white paper.

[8] Newcomb, N. (Chief Executive Officer, TSRI): *Interview* by R. Sree K V, 25 July 2026. Open Mainframe Project Summer Mentorship 2026. Recorded and transcribed; quoted with permission.

[9] Swimm: *A blog article is not a product. No, Claude Code can't understand COBOL applications*. Swimm blog, February 2026. Vendor test; see Appendix B. Link

[10] Sree K V, R.: *Discovery Is Not Migration: Where AI Actually Stands for Mainframe Modernization*. Open Mainframe Project blog, Summer Mentorship 2026, July 2026. Link

[11] Walter, S.; Dickens, S. (HyperFRAME Research): *Mainframe Modernization in the AI Era: How Generative AI Is Transforming Legacy Systems into Intelligence Hubs within the Enterprise AI Stack*. Research brief, July 2026. Link

[12] BMC Software: *20th Annual BMC Mainframe Survey Reveals Confidence, Generational Shift in Views on Mainframe Development*. Press release, 16 September 2025 (survey fielded 31 March–15 April 2025; more than 1,000 respondents). Link

[13] BMC Software: *Trust Powers AI on the Mainframe*. Press release, 1 September 2026 (2026 BMC Mainframe Survey; more than 1,300 respondents). Link

[14] Rabbi, F.; Saha, S. K.; Yang, J.: *Beyond Translation Accuracy: Addressing False Failures in LLM-Based Code Translation*. arXiv:2605.02195, May 2026. Link

[15] METR: *Measuring the Impact of Early-2025 AI on Experienced Open-Source Developer Productivity*. 10 July 2025. Link Update: We are Changing our Developer Productivity Experiment Design, 24 February 2026. Link

[16] Cui, K. Z.; Demirer, M.; Jaffe, S.; Musolff, L.; Peng, S.; Salz, T.: *The Effects of Generative AI on High-Skilled Work: Evidence from Three Field Experiments with Software Developers*. Management Science, online 27 February 2026. Link

[17] DORA (Google Cloud): *2025 State of AI-assisted Software Development Report*. 2025. Link

[18] Dickens, S. (Chief Executive Officer and Principal Analyst, HyperFRAME Research): *Interview* by R. Sree K V, August 2026. Open Mainframe Project Summer Mentorship 2026. Recorded and transcribed; quoted with permission.

[19] Britton, D. (Tech Marketing Leader and Advisor; CMO, PopUp Mainframe): *Interview* by R. Sree K V, 20 July 2026. Open Mainframe Project Summer Mentorship 2026. Recorded and transcribed; quoted with permission.

[20] Brach, G. (Chief Executive Officer, Phase Change Software): *Interview* by R. Sree K V, 17 July 2026. Open Mainframe Project Summer Mentorship 2026. Recorded and transcribed; quoted with permission.

[21] Anthropic: *I want to opt out of my prompts and results being used for training models*. Anthropic Support, section "For Anthropic Commercial Offerings". Link

[22] Britton, D.: *Mainframe, AI, and Modernization – Today's Perspectives*. PopUp Mainframe, 29 July 2026. Link

[23] Korn Ferry Institute: *Age and Tenure in the C-Suite: Korn Ferry Institute Study Reveals Trends by Title and Industry*. Press release, 14 February 2017 (study of the 1,000 largest US companies, conducted in late 2016). Link

[24] bloop; Zorse Project: *COBOLEval*, a transpilation of the HumanEval benchmark from Python into COBOL. GitHub repository. Link

[25] Chen, M.; Tworek, J.; Jun, H.; et al.: *Evaluating Large Language Models Trained on Code*. arXiv:2107.03374, July 2021. Link

[26] Michels, D. L.; Abu Ghazaleh, M.; Lazzari, F.; Kassem, N.; Klein, J.: *Vibe Coding: Practice, Performance, Productivity, and Risk – A State-of-the-Art Review*. arXiv:2608.20446, 20 August 2026. Reports Faros AI telemetry. Link

About this paper

Our authors

This document was developed under the Open Mainframe Project's Summer 2026 Mentorship program and prepared for the Mainframe Modernization Working Group.



Ramana Sree K V | Mentee @ LFX 2026 Summer Mentorship

Final-year undergraduate researching AI and mainframe application modernization. Author of this paper: designed and conducted the four-interview series, reviewed the measured and analyst literature, and developed the evidence-gated modernization framework.



Misty Decker | Project Mentor

Mentor for this project in the Open Mainframe Project's Summer 2026 Mentorship. Opened her professional network for the interview series, joined interviews, and guided the paper's method and evidence discipline. She is not cited as a research source.

Special thanks

This work would not have been possible without the time and candour of the four practitioners and analysts who were interviewed for it, and who consented on record to being quoted:

- **Gary Brach**, Chief Executive Officer, Phase Change Software
- **Derek Britton**, Tech Marketing Leader and Advisor, and CMO at PopUp Mainframe
- **Nicolas Newcomb**, Chief Executive Officer, TSRI
- **Steven Dickens**, Chief Executive Officer and Principal Analyst, HyperFRAME Research

Thanks also to the Open Mainframe Project and the Linux Foundation for the Summer Mentorship 2026 program.

Thanks also to **Nithin Sai Jalukuru** for his peer review of this paper.

About the modernization Working Group

The modernization Working Group, part of the Open Mainframe Project, intends to be a focal point for thought leadership around what it means to modernize mainframe applications.

We are an open group passionate about knowledge sharing, leading constructive discussions, challenging status quo, exploring creative solutions, and generating contents that will help the community in their journey.

Learn more about the working group at:

https://openmainframeproject.org/our-projects/working-groups/modernization-working-group/