AGI ALPHA FLYWHEEL ∞

BUILD THE NEXT ADVANTAGE

From Useful Work to Expanding Productive Capability

An executable study of example-driven capability construction, fresh work, complete costs and financeable succession.

SUCCESSOR Ω HARNESS

ACTIVATE. EVOLVE. SCALE.

TRANSFORMATIONAL INTELLIGENCE ∞

Applicable $AGIALPHA utility is documented by source and route.

Vincent Boucher · MONTREAL.AI

25 October 2026 | Study NADV-2026-01

Local research executed. Monetary outcomes modeled.
Persistent authenticated operation and real financial succession not run.

ORIGINAL RESEARCH CONTRIBUTION

Useful work can construct its next input capability. The strongest incumbent still matters.

Abstract

Can automated work support the construction of capabilities that make subsequent work more useful after their complete costs? We extend the existing AGI ALPHA Workbench and Autonomous Work Economy quote workflow with a finite, example-driven constructor of executable input adapters. The unchanged quote specialist already produces useful comparison packages, but requires canonical offers. The new constructor infers column and unit transformations from budgeted examples, qualifies each adapter on fresh work, and reuses admitted programs on later missions. An independently implemented Python verifier checks input semantics as well as the final cart and report. A frozen study covers 24 held-out schemas, 96 unique production missions, five methods and 24 separate boundary probes. Every method accepts 96/96 nominal missions; conventional active selection and a current LLM-assisted constructor each use 29 examples, versus 61 for ordinary reuse. A learned construction recipe also uses 29 and requires 31 development examples: improvement beyond the strong incumbent and sustained acceleration are not established. Complete-cost scenarios yield an active-method cost of $1.211784 per accepted report in the base case, rising to $15.863884 and $650.040140 under lower utilization and shorter recovery horizons. These are modeled costs, not bills. Thirty nontransactional financing paths expose affordable, loss-making and delayed-payback cases for a hypothetical future capability. Actual collections and available earned surplus remain unknown. The contribution is an inspectable, source-bound construction-to-delivery loop with explicit technical qualification and economic non-promotion, rather than a claim of autonomous algorithmic invention or self-financing operation.

New capability	24 example-derived programs prepare previously unsupported raw-table inputs for the existing quote workflow. Each is qualified before three subsequent fresh reuses.
Strong comparison	Active selection: 29 examples. Ordinary cached scripts: 61. Learned recipe: 29 plus 31 training examples. Current LLM-assisted constructor: 29; full model cost unknown.
Economic decision	Keep the active specialist. Hold the extra learned recipe. Human-work savings remain duration thresholds; future financing remains simulation.

Scope of originality. We contribute the AGI ALPHA-specific preparation contract, executable construction and qualification mechanism, source-bound evidence chain, and separated complete-cost admission experiment. Programming by example and active example selection are established ideas [1]. We do not claim their invention, a new quote-optimization algorithm, a superior general LLM, or a demonstrated recursively improving enterprise.

Contents

Research contribution and abstract 2

1 Research question and immediate customer value 4

2 Foundations and the limitation addressed 4

3 Useful deliverable and acceptance contract 5

4 An executable construction loop 6

5 Prospective evaluation and comparators 7

6 Results: useful construction, a strong tie, no acceleration 10

7 Complete-cost economics of accepted work 12

8 Financeable succession: affordability and payback windows 15

9 Ecosystem roles, authority and token boundaries 16

10 English/French demonstration and reproducible companion 18

11 Limitations and exact execution status 19

12 Conclusion 20

A Source identities and preservation 21

B Failure, amendment and cost accounting record 21

C Research responsibilities and next decisive experiments 22

Reading the evidence. "Accepted" means that the complete synthetic research deliverable passed the declared independent verifier. It does not mean customer acceptance, production authorization or payment. "Constructed" means that a program was derived by the declared constructor from allowed examples. "Modeled" and "simulated" monetary quantities never become observed receipts.

1 Research question and immediate customer value

The AGI ALPHA FLYWHEEL asks whether useful automated work can create resources for capabilities that make subsequent work more valuable, less costly or newly possible. A stronger hypothesis is that a retained capability improves the construction of the next capability. Both hypotheses need a complete causal record: useful output, an observed limitation, actual construction, fresh qualification, later reuse, costs and an affordable next decision. A favorable description of an existing solver does not supply this record.

The present offering is a bounded quote-comparison service. A customer supplies item quantities, supplier offers, pack sizes and prices, explicit exchange rates, shipping charges and a deadline. The service returns a recommended purchase basket, a readable report, an analysis record explaining exclusions and exact costs, and evidence linking the inputs to the calculation. It could replace the clerical preparation, arithmetic, comparison and report assembly in a competent human's quotation worksheet. It does not negotiate, certify suppliers, infer unstated commercial terms, procure goods or authorize payment. Equivalent human work must have the same deliverable and acceptance standard; comparisons with an entire procurement function would overstate replacement.

The existing service is useful when offers already have its canonical fields. Its unresolved preparation boundary is practical: supplier tables differ in column identity, units and money notation. The inspected TRANSFORMATIONAL INTELLIGENCE guide compares same-named columns, and the AWE quote adapter expects normalized offer objects [2, 3]. We therefore select schema preparation as the next capability. We retain the established quote specialist and its existing delivery runtime. A new domain or a replacement optimizer would add little evidence about this actual bottleneck.

1.1 Three separate estimands

A. Human-work economics asks whether the delivered package can cost less than equivalent competent work without AI. Because matched human duration is unavailable, we compute explicit break-even durations from sourced labor rates and stated delivery assumptions. We do not invent a human completion time.

B. Incremental Flywheel benefit asks whether retained executable capability improves fresh production beyond ordinary reuse, a competent builder and established specialist methods, after training, construction, failure and maintenance. A cached adapter can be useful even if a conventional active selector is the best way to construct it. Retention alone does not establish a distinctive Flywheel advantage.

C. Financeable succession asks whether available earned resources, after costs, obligations and reserves, can fund beneficial further construction within its useful life. Affordability and value are jointly necessary. A profitable hypothetical service does not prove that the proposed next capability works; a useful capability does not prove that cash is available when needed.

Our primary technical unit is a schema-level construction problem followed by four fresh missions. Our economic unit is an accepted complete report. Those units must not be conflated with a string-conversion kernel, an invoice, a model token or an assumed token-market return.

2 Foundations and the limitation addressed

We inspected the four requested repositories, the coordination record in issue #4850, source artifacts and earlier manuscripts. Table 1 states what the prior work supports and what this edition adds. Original manuscripts, study outputs, URLs and source histories remain unchanged. Several source and software lanes remain unmerged; publication of a passive paper is distinct from software qualification [11].

The source-assurance workflow remains an important commercial foundation: it checks approved public source

Table 1: Evidence lineage [4, 5, 6, 7, 3]. Counts are prior-edition findings, not replications by this study.

Foundation	Established within its scope	Remaining limitation
Economics of Autonomous Work	36/36 fresh exact-reconciliation jobs; conditional labor thresholds.	Matched human duration and complete deployed cost unknown.
Capability construction	42/42 primary jobs; actual parent/child calls; specialist comparisons.	Repair loses on 12/12 fresh cases; three cycles admit no improvement. Evidence adapter was externally authored.
Productive Capital Engine	72 scoped results; assurance, planning and dossier connection; cash and resource checks.	Child fresh-change CPU 4.069055 s versus 4.032727 s for ordinary caching; improvement HOLD.
Economic Engine	144 financial paths separate collections, reserves, failed candidates and finite-life payback.	No primary path buys a new candidate or capacity; simulation does not construct a method.
Autonomous Work Economy	16 accepted packages and 8 intended refusals; separate quote transfer 8/8 for successor and MILP, 2/8 for initial capsule.	Capability templates and solver composition were authored in advance; canonical input preparation assumed.
This edition	Example-derived input programs; first fresh qualification before reuse; full alternative and cost record.	Synthetic bounded domain; no superiority to active selection, measured customer savings or observed self-financing.

identities, inventories discrepancies and produces actions and reports. It does not certify behavior or security. Its exact-key/hash incumbent remains appropriate. Its current improvements are safety engineering, not automatically autonomous invention. Likewise, planning capabilities and the earlier repair specialist provide real reusable components without establishing a superior learned builder [5, 6].

We use the released Alpha `createWorkbench(adapters)` extension seam, the unchanged AWE quote normalization and supplier-subset solver, existing source/mission identities, pack integrity and unfunded Jobs compilation. We import the owner's exact `alpha-automated-costs/v1` and `alpha-shadow-ledger/v1` implementations. The new module is a bounded adapter and constructor; it is not another runtime, scheduler, accounting schema or publishing system.

The contribution therefore addresses a narrower and previously unresolved link: *can evidence from a useful workflow produce a new executable preparation capability, be checked independently on fresh work, and be retained with a defensible economic decision?* It tests this link even when the next learned construction method deserves rejection.

3 Useful deliverable and acceptance contract

3.1 What the user receives

Each accepted pack contains five useful artifacts: `quote-report.md`, `recommended-cart.csv`, `quote-analysis.json`, `constructed-adapter.json` and `prepared-offers.json`. The first three come from the existing quote workflow; the last two expose how raw input became canonical offers. The Workbench supplies source and mission bindings, integrity digests, evaluation labels, a chronicle and seven unfunded Jobs drafts. Draft compilation does not submit jobs or move money.

For every requested item, the basket selects one eligible supplier offer and enough integral packs to cover demand. It includes the applicable rational exchange-rate conversion, exact declared rounding, supplier shipping charged once, and deadline exclusions. The independent oracle enumerates feasible carts using a separate Python implementation

and verifies the optimum and declared constraints. Equal-cost alternative baskets are permitted if they satisfy the same exact acceptance standard. The service does not infer exchange rates or interpret open-ended free text.

The complete-output check also verifies every prepared offer against evaluator-owned canonical input, the CSV and report totals, and the analysis. This input check matters: two solvers that share the same incorrectly normalized input can agree perfectly and still deliver the wrong result. Workbench integrity inspection alone is insufficient to establish semantic correctness.

3.2 Automated operating configuration and residual work

The local configuration performs typed preparation, construction, planning through the existing adapter, execution, verification, artifact packaging and explicit refusal. It caches only a qualified program under the complete schema contract. A first fresh qualification failure prevents reuse; the expected answer is never fed back to repair a held-out program. Atomic journal writes preserve completed events, and bounded cancellation is available. The demonstration offers inspection, downloads and a restartable local run.

Correction is deliberately constrained. Unsupported or ambiguous input is refused with a reason; it is not silently coerced. A user or operator must resolve an unknown schema, provide a reviewed example, correct a missing exchange rate or repair a contradictory source record. The model charges example preparation, exceptions, retries, maintenance, setup and customer attention explicitly. Remote recovery, scheduled intake, network reconnect, browser-close persistence and clean installation on an approved execution profile were not run. Atomic local output is not evidence of those service properties.

Concrete example, not a sale. H01 needs two selected worked examples to identify its adapter. Its first fresh synthetic basket totals 858,559 USD minor units; the next totals 2,824,085. Both complete packages pass the separate Python check. They are different purchase requests, so the difference in totals is not a saving. The reusable asset is the exact mapping and interpretation program, not either basket's answer.

4 An executable construction loop

4.1 Program space and declared permissions

A schema task has 6–10 distinct headers and a pool of 16 unlabeled rows. It exposes an oracle for a reviewed row's six canonical values: item, vendor, pack units, pack price in minor units, currency and lead days. Each program assigns every canonical field a source column and one allowlisted operation. It is inert JSON interpreted by a finite grammar; no input-supplied source code is evaluated.

Text is trimmed without changing interior content. Currency is an explicit three-letter code. Pack counts accept integers, declared suffixes such as "units" or "pièces", or an `xN` prefix. Lead time accepts integers, declared day suffixes or weeks multiplied by seven. Prices accept bare nonnegative integer minor units, or nonnegative decimal point/comma notation with exactly two fractional digits. Only the decimal forms permit an optional leading $ or euro symbol or a trailing three-letter currency code, never both; $ must agree with USD and the euro symbol with EUR. Bounds, identity collisions and currency conflicts are checked. Thousands separators, approximate currency inference, arbitrary transformations and undeclared semantics are outside scope.

The researcher supplies this grammar, constructor, tools, budgets, acceptance standard and experimental infrastructure. The system chooses the mapping and operation for each actual schema from permitted evidence. No superior schema-specific mapping is manually inserted as a candidate. Constructed artifacts, alternatives eliminated, labels

queried and failures are recorded. This is *new executable construction within an authored grammar*; the quote solver is adoption and reuse; the Workbench seam is integration. None is presented as autonomous algorithmic invention.

For field f, let V_f be the finite set of candidate column–operation pairs valid on the unlabeled pool. After querying row i with canonical value y_{if}, the constructor updates

$$V_f \leftarrow \{p \in V_f : p(x_i) = y_{if}\}. \tag{1}$$

It emits a program only after at least one example and exactly one survivor for every field. Empty spaces, unresolved ambiguity, cancellation or exhausted budget produce refusal. The limits are eight labels and 200,000 candidate-evaluation operations per construction. Selection operations are separately counted and included in measured constructor CPU/wall time. Equal caps do not imply equal consumed resources.

4.2 Ordinary and learned construction policies

The ordinary builder queries rows in their declared order. Conventional active selection chooses an unqueried row maximizing normalized disagreement across fields. If n_{fiv} candidates predict value v, its score is

$$S(i; w) = \sum_f w_f \left(1 - \frac{\sum_v n_{fiv}^2}{\max(1, |V_f|^2)}\right). \tag{2}$$

The established active policy uses $w_f = 1$. The experimental retained recipe measures unresolved alternatives after development examples, $d_f = \sum_{t,e} \max(0, |V_{f,t,e}| - 1)$, and derives

$$w_f = 1 + \text{round}\left(7 d_f / \max(1, \max_g d_g)\right). \tag{3}$$

This one retained construction capability is then evaluated on unseen schema problems. Its weights are item 5, vendor 1, pack units 2, price 8, currency 1 and lead days 1. They are computed from traces rather than set after looking at held-out performance. The recipe is a hypothesis about better example selection; transfer and repayment are tested separately.

4.3 From mission evidence to the next mission

Figure 1 shows the execution and decision boundaries. The companion's connected replay first runs a canonical seed job through the unchanged adapter and verifies it. A raw-table request then encounters the actual preparation boundary (the existing workflow reports `OFFER_LIMIT` without canonical offers). The study constructor derives H01's program, the first fresh mission qualifies it, the next reuses it, and the shared ledger records the unknown real financial position. This is a scripted, after-outcome replay connecting executed study cases, not an additional independent experiment or evidence of autonomous mission-family selection.

5 Prospective evaluation and comparators

5.1 Freeze, separation and independent verification

Before confirmatory outcomes, the protocol freeze bound 371 files: the grammar, source modules, cost assumptions, input commitments, primary criteria, evaluator and sealed data. It was recorded at 21:02:05 UTC on 25

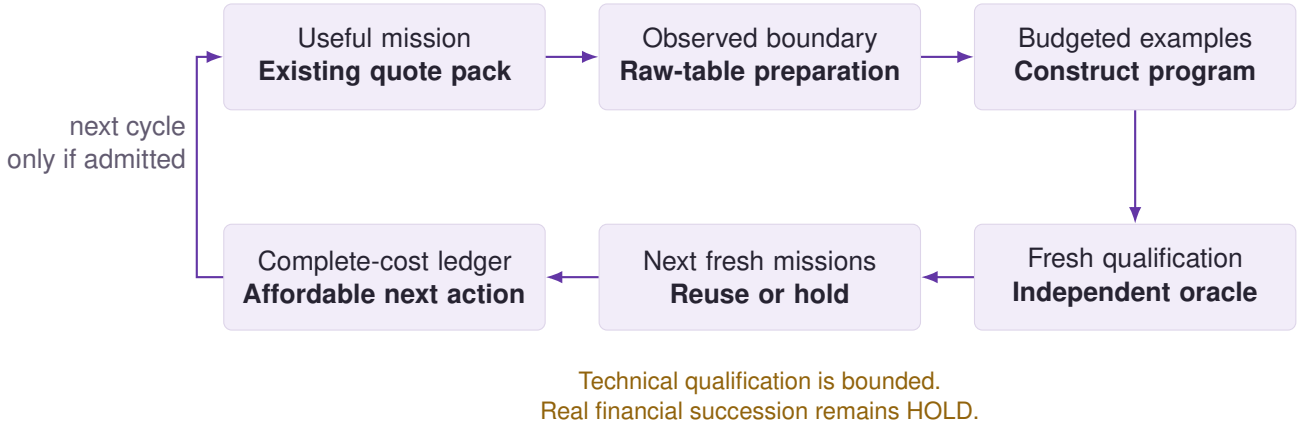


Figure 1: The executed connection. New scientific acceptance is separate from existing pack integrity. Hypothetical financing scenarios cannot clear the final real-money gate.

Table 2: The intended journey mapped to concrete actions and evidence. The execution order loops through useful work; names do not imply additional unexecuted reasoning modules.

Stage	Input, action and artifact	Measurable contribution
Identify	Supported quote input; select complete report contract; save seed input.	Defines the accepted useful unit.
Out-Learn	Verified seed and failed raw-input attempt; record preparation boundary.	One observed contract limitation, not a benchmark victory.
Out-Think	Boundary and permitted grammar; choose finite example-driven preparation.	Search limited to six fields, eight labels and explicit operations.
Out-Design	Allowed examples; execute constructor; save program and elimination trace.	24 programs per retained builder; observed acquisition cost.
Out-Strategise	Frozen comparison and shared costs; keep active incumbent.	Extra learned recipe not economically promoted.
Out-Execute	Fresh rows and program; run unchanged quote specialist; verify pack.	96/96 accepted per method.
Out-Evolve	Qualified program and ledger; reuse on next jobs; assess next action.	72 subsequent reuses per caching method; real finance HOLD.

October 2026. The freeze identifier begins `d0d0f94598c86f3b`; the full identifier and file hashes are in the companion. The confirmation runner checks every frozen file before running. The LLM constructor has its own subsequent source/program freeze before any production evaluation.

Twelve development schemas cover plain minor-unit prices, suffix/dot notation and prefix/comma notation, four each. Twenty-four held-out schemas contain four instances of each of six families: the same three types plus weekly lead times, currency affixes and mixed alternative views. The last three are unseen structural compositions within the disclosed grammar, not arbitrary new language features. Header identities and exact configurations are disjoint. Four fresh production missions per held-out schema yield 96 unique jobs. All use synthetic, nonpersonal data; no market sampling is implied.

The builder sees the task's unlabeled example pool and only explicitly requested canonical labels. It does not receive production expected outputs. Evaluation ownership is separate within the collaborating research session. The Python verifier independently implements exact arithmetic and Cartesian cart search, checks normalized input truth, and imports neither the candidate nor the JavaScript quote solver. It was exercised on existing AWE cases and deliberately mutated outputs before freeze. This is implementation independence, not external institutional peer review or cryptographic isolation from the supervising agent.

The first fresh mission is the technical qualification trial. Only after its independent pass may the program be reused on the next three. Confirmation is frozen and nonadaptive: failures do not change the candidate. Every case remains in the denominator. An incorrect accepted output violates the primary safety criterion; refusals reduce delivered coverage. Twenty-four separate boundary probes test expected rejection, rather than being counted as useful deliveries.

5.2 Five methods with legitimate improvement opportunities

1. **Rebuild ablation.** The ordinary sequential constructor runs anew for every mission. This exposes construction repetition; it is not the strongest comparator.
2. **Ordinary reuse.** The same constructor uses schema-keyed reusable programs after qualification. This gives conventional caching and scripts their legitimate benefit.
3. **Retained construction recipe.** The same grammar, constructor and cache use the learned weights in Equation 2. Development is charged.
4. **Active specialist.** Conventional maximum-disagreement selection has the same grammar, label cap, candidate cap and cache, without recipe training. This is the principal strong deterministic comparator.
5. **Current LLM agent with standard tools.** A separate clean-context assistant authors and runs a Python finite-grammar constructor using allowed public examples and a bounded oracle CLI. It may construct reusable capabilities and receives a grammar reviewer. Its programs are frozen before production feedback and delivered through the same runtime and verifier.

The current LLM comparator is an actual same-session tool-assisted execution, not a hypothetical API agent or a static result attributed to a model. Its action log, source revisions, selected labels, programs, parser checks and recorded-label replay are retained. A limited incidental sibling-summary metadata exposure is disclosed; no production inputs, expected results or candidate implementation were opened before its freeze. Model inference time, provider token counts and billing are unexposed. Therefore it supports output and example-efficiency comparison but not a matched-dollar or externally blinded LLM benchmark. Its Python operation counters are also not a portable instruction-equivalent measure against JavaScript.

All methods use the same existing quote specialist downstream. A different optimizer would not resolve preparation and would confound this study. AWE already compared its quote method with conventional MILP [3]. Here the relevant established specialist is active example selection; we retain it when it wins.

5.3 Primary outcomes and uncertainty

Primary outcomes are accepted complete outputs, false acceptance, refusal and example acquisition. We also record constructions, hypothesis evaluations, selection operations, CPU/wall spans, artifact bytes, reuse and unknown monetary fields. All four deterministic arms are interleaved with rotating order. The LLM production evaluation follows its independent freeze and is not timing-interleaved with them.

The paired query analysis uses schemas as clusters, with all four missions linked. We resample four schemas within each of the six family strata, using 10,000 bootstrap draws and seed 20261025. The interval describes this generated workload, not market demand or universal schema performance. Tiny single-host CPU differences are descriptive; no service-pricing or general latency claim rests on them. Complete process CPU is unknown because Python startup/teardown CPU was not instrumented; complete verification wall time includes process launch.

Table 3: All primary methods on the same 96 unique fresh jobs. Counts across columns are repeated evaluations, not independent customer jobs. Additional learned-recipe training is separate.

Method	Accepted	Builds	Labels	Extra training	False accepts
Rebuild	96/96	96	244	0	0
Ordinary reuse	96/96	24	61	0	0
Retained recipe	96/96	24	29	31	0
Active specialist	96/96	24	29	0	0
Current LLM-assisted	96/96	24	29	See text	0

The LLM constructor used 15 selected examples during its 12-schema public development run and validated all 192 public development labels. Its source-authoring/model cost is unknown; these counts do not make it cost-free.

6 Results: useful construction, a strong tie, no acceleration

All 480 method–job evaluations pass on the 96 unique production missions. There are no nominal refusals or independently rejected outputs. Each caching method qualifies 24 programs and uses them on 72 subsequent jobs. The untouched canonical-input workflow has no raw-table adapter; the new programs expand the input forms it can deliver on. This establishes scoped input-capability construction, not a quality advantage over the capable alternatives, which also deliver all jobs.

The active policy uses 32 fewer examples than ordinary reuse, a 52.46% reduction from 61. The retained policy makes the same reduction but also consumes 31 development examples. The paired retained-minus-ordinary difference is -1.333 examples per schema, with a stratified bootstrap 95% interval $[-1.833, -0.833]$. Retained-minus-active is exactly zero on every schema and has interval $[0, 0]$. The LLM-assisted constructor independently selects 29 examples, at most two per held-out schema, and produces programs whose recorded-label replay exactly matches its frozen results.

Table 4: Family-level acquisition totals. Each family has four held-out schemas and 16 fresh missions per method. All corresponding deliveries are accepted.

Family	Ordinary reuse	Retained recipe	Active specialist
Plain minor units	14	6	6
Suffix / decimal point	12	6	6
Prefix / decimal comma	4	4	4
Weekly lead times	6	4	4
Affixed currency	8	4	4
Mixed alternative views	17	5	5

6.1 Resource record and meaningful failures

The deterministic rebuild, ordinary, retained and active methods perform respectively 44,420, 11,105, 10,866 and 10,866 candidate evaluations. Retained and active each record 9,053 selection operations. Constructor CPU totals are 91.378, 19.408, 23.021 and 25.772 ms. Whole recorded mission wall totals, including verification and packaging, are approximately 5.857, 5.785, 5.785 and 5.881 seconds over 96 reports. These small one-run differences do not establish a general speed ranking.

The LLM-authored CLI records 38,591 conservative parse/comparison budget units, 37,272 selection operations and 766.133 ms of per-schema constructor CPU. Its combined recorded construction-and-delivery wall time is about 7.023 seconds, including 0.763 seconds of construction and 6.259 seconds of delivery, verification and packaging.

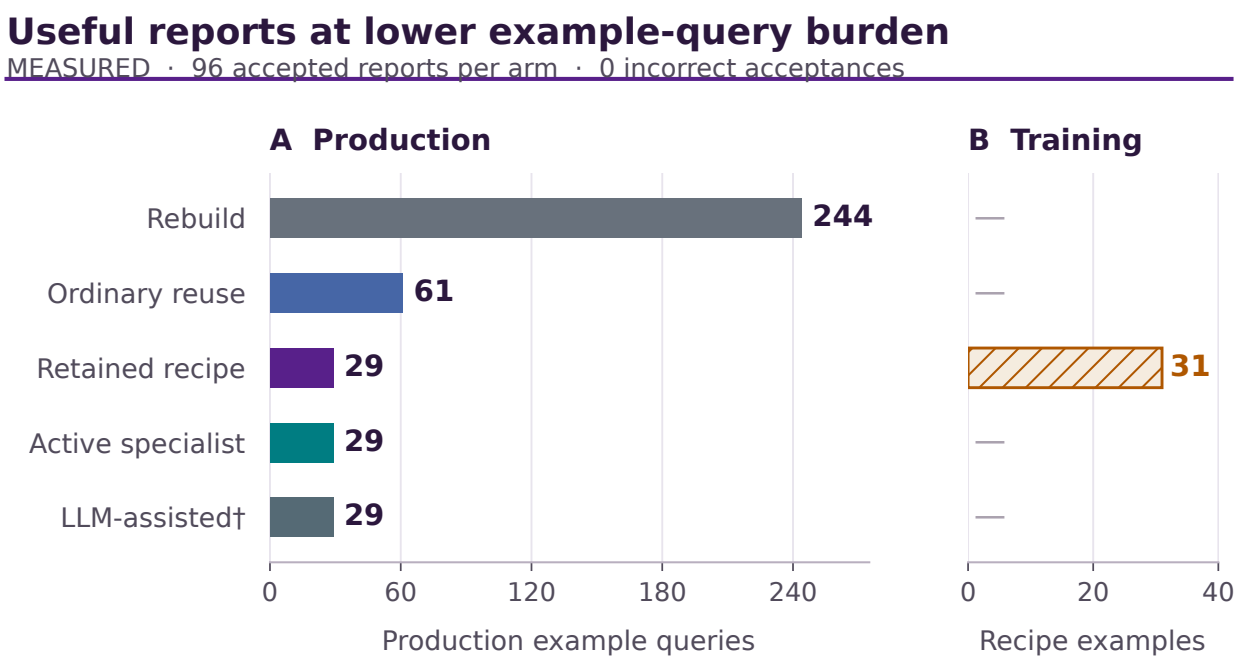


Figure 2: Example acquisition for useful programs. The extra learned-recipe training is visible. Conventional active selection matches both the retained recipe and the actual LLM-assisted constructor on production labels. The LLM's authoring and model billing remain unknown.

These omit model inference and interactive authoring, and its counters differ in granularity. We therefore make no inference that either language, constructor or LLM is economically faster from these figures.

All 24 unique boundary probes reject as required across the three deterministic selection policies, yielding 72 repeated checks. The six groups cover indistinguishable columns, conflicting currency affixes, identity collisions after normalization, malformed or out-of-bounds values, schema drift, and unsupported semantic identities or exchange rates. Four ambiguous cases consume the eight-label budget and refuse. The separate LLM constructor was not evaluated on those four construction-ambiguity probes; the 72 checks must not be attributed to all five constructors. Such refusals are correct behavior, not successful useful work or improved productive capability. A separate control suite checks budget exhaustion, cancellation and literal grammar behavior.

No nominal candidate failed after freeze, and none is omitted. Pre-freeze development corrections, all three development executions, source snapshots and failed checks remain in the package. The three executions contain 93 query calls but only one 31-example training episode; they are not three independent gains. Reporting amendments preserve the initial unstratified bootstrap output and the auxiliary CPU report that omitted packaging. Corrected reporting uses the prespecified family strata and all recorded CPU spans. Neither correction changes candidate source, labels, accepted outputs, cost tiers or financing decisions.

6.2 What improved, and what did not

Better task execution is supported within scope. A qualified retained adapter makes a new input schema executable on subsequent fresh missions. Ordinary cache reuse alone removes 183 repeated acquisition queries

relative to rebuild. Active acquisition further removes 32 relative to sequential ordinary reuse. These are real finite-workload resource differences.

A better retained builder beyond the strong alternative is not supported. The learned weights transfer to new schema problems, but the untrained active specialist reaches the same programs with the same production acquisition. Its extra training and maintenance cannot be justified by the measured primary benefit. The correct method-selection result is to keep the incumbent active policy.

Sustained acceleration is not established. There is one learned construction recipe, one held-out construction episode and a finite set of repeated missions. No sequence of successively more capable builders, compounding rate of invention or persistent enterprise has been demonstrated.

7 Complete-cost economics of accepted work

7.1 Shared contract and provenance of each quantity

The economic analysis calls the unchanged shared cost and shadow-ledger modules from the inspected source-assurance owner's branch [8]. These distinguish measured, sourced, estimated and unknown facts. The cost contract uses integer micro-dollars and exact rational intermediates; the shadow ledger records integer currency minor units (USD cents here). The scenario adapter rounds micro-dollar costs upward when converting to ledger cents. We preserve separate price, hypothetical collections, costs, obligations, reserves, customer attention and available surplus. No token issuance or customer saving enters provider revenue.

Measured inputs include report acceptance, example counts, failures and recorded execution spans. **Sourced** labor anchors are June 2026 U.S. private-industry employer compensation of $36.73/hour for office/administrative work and $73.69/hour for professional/related work; a separate May 2025 procurement-clerk median wage of $24.32/hour excludes benefits [9, 10]. These sources do not supply a matched duration for this deliverable.

Assumed inputs include equivalent reviewed-example minutes, runtime-session allowance, utilization, operational acceptance, retries, exceptions, maintenance, authoring/setup, useful life and customer attention. All prices, demand, collection behavior and future capability benefits in the finance experiment are simulated. **Unknown** inputs include actual full cost, actual human duration, actual customer effort, model-authoring billing, actual revenue and earned investable surplus. A deterministic runtime with zero external provider calls does not imply that the research or service costs zero.

Let q denote requested jobs per month, a the operational accepted fraction, x all variable expense per request, F recurring shared expense per month, and I_k/L_k amortized setup or construction expense. The cumulative cost tiers are

$$C_{\text{marginal}} = x/a, \tag{4}$$

$$C_{\text{recurring}} = (qx + F)/(qa), \tag{5}$$

$$C_{\text{full}} = (qx + F + \sum_k I_k/L_k)/(qa). \tag{6}$$

Execution-only cost is a narrower preceding tier. The tiers are cumulative and must not be added together. Customer attention stays visible outside provider cost. The operational acceptance assumptions deliberately differ from this small synthetic experiment's 100% observed pass rate.

The base setup allowance consists of 16 engineering hours at the professional rate of $73.69/hour, four review hours at the office/administrative rate of $36.73/hour, and $100 for model authoring, totaling $1,425.96. Runtime-session allowance covers the complete local run, including verification process launch and packaging; it is not a

Table 5: Prespecified cost assumptions. The scenario names do not represent measured probabilities or confidence intervals. All monetary inputs are assumed except the cited labor rates.

Input	Base	Conservative	Stress
Capacity / requested reports per month	1,000 / 500	1,000 / 100	1,000 / 10
Operational accepted fraction	98%	90%	65%
Attempts per request	1.02	1.15	1.50
Runtime-session allowance / attempt	$0.03	$0.10	$0.30
Equivalent reviewed-example minutes	2	5	10
Exception rate $\times$ attention minutes	$1\% \times 10$	$5\% \times 20$	$20\% \times 45$
Shared platform expense / month	$60	$120	$180
Maintenance / business minutes per month	120 / 60	360 / 180	960 / 480
Common setup cost	$1,425.96	$3,541.44	$7,482.88
Setup recovery months	12	6	3
Extra recipe engineering minutes	30	120	480
Extra recipe upkeep minutes / month	5	20	60
Recipe useful life, months	6	3	1
Customer attention minutes / report	3	5	15

CPU-only price. The eight native stages receive accounting allocations: preparation 15%, planning 5%, execution 20%, verification 25%, correction 10%, packaging 10%, reporting 5% and recovery 10%. These are declared cost allocations, not measured tariffs. Retries, failed bounded search, exceptions and setup validation are included under the explicit assumptions.

The example-work extrapolation preserves the study mix of four reports per new schema: production queries divided by 96 requested reports. It does not claim that an already cached schema requires labels indefinitely. More reuse, more schema drift or a different new-schema arrival rate would change the result. The current LLM comparator remains outside the immutable four-arm monetary model because its full authoring and inference costs are not observed.

Table 6: Exact cumulative modeled USD per accepted report. The shared implementation rounds upward to micro-dollars.

Method	Scenario	Execution	Marginal	Recurring	Full
Active	Base	0.031225	0.471090	0.969274	1.211784
Ordinary	Base	0.031225	0.887530	1.385714	1.628224
Rebuild	Base	0.031225	3.269046	3.767229	4.009740
Retained	Base	0.031225	0.471090	0.981806	1.249769
Active	Conservative	0.127778	1.835329	9.305662	15.863884
Ordinary	Conservative	0.127778	2.968970	10.439303	16.997525
Rebuild	Conservative	0.127778	9.451985	16.922318	23.480540
Retained	Conservative	0.127778	1.835329	9.578588	17.034461
Active	Stress	0.692308	12.013474	266.302704	650.040140
Ordinary	Stress	0.692308	15.152787	269.442018	653.179453
Rebuild	Stress	0.692308	33.105752	287.394982	671.132418
Retained	Stress	0.692308	12.013474	277.639628	781.314243

7.2 A. Human-work economics: thresholds, not invented savings

For hourly human cost r and provider full cost C, equivalent-work break-even duration is $t^* = 60C/r$. If customer attention costs E, total-resource break-even is $60(C + E)/r$. A buyer instead compares a price P, giving $60(P +$

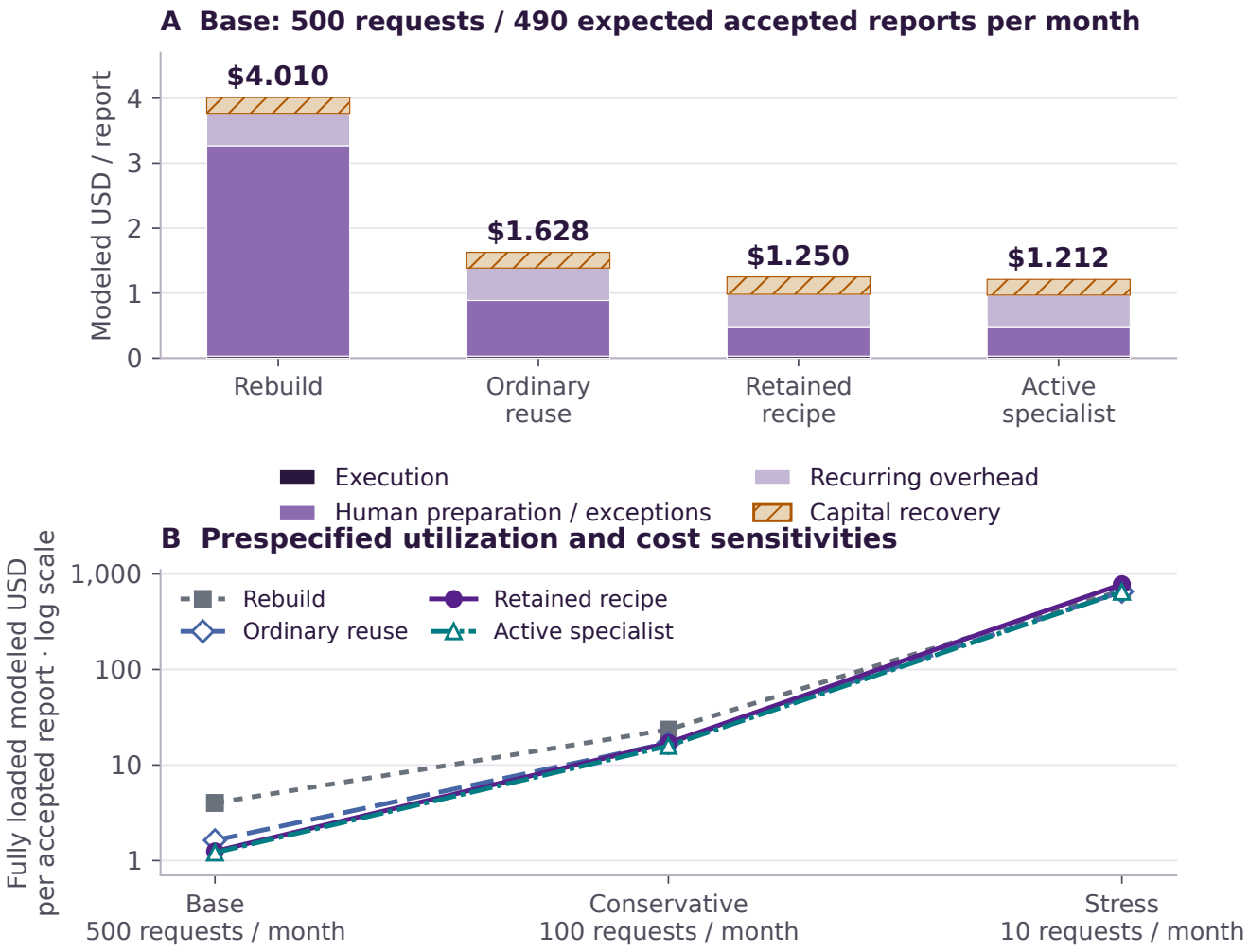


Figure 3: Complete cost changes the decision. Base costs include preparation, exception work, recurring overhead and capital recovery. Lower utilization and shorter recovery horizons dominate the stress case. All amounts are modeled USD; no invoice or customer sale was measured.

$E)/r$. These are distinct questions.

At $36.73/hour, the base active provider cost of $1.211784 corresponds to 1.9795 human minutes. Three assumed customer minutes at that same rate add $1.8365, producing a total-resource threshold of 4.9795 minutes. At an assumed $15 price, the buyer threshold is 27.5031 minutes. Thus a low modeled provider cost is not automatically a large customer saving at any chosen price.

For provider cost alone, the conservative and stress thresholds at the office rate are 25.914 and 1,061.868 minutes. At the professional compensation rate, base/conservative/stress thresholds are 0.987, 12.917 and 529.277 minutes;

at the procurement wage-only rate, 2.990, 39.138 and 1,603.717 minutes. The lower wage concept should not be silently mixed with employer total compensation. The companion includes price sensitivities of $1, $5, $15 and $50.

These thresholds make today's supported offer understandable: the exact local workflow can already deliver useful comparison packages, and its economics can be attractive if actual equivalent human work exceeds the relevant threshold under a viable utilization and attention regime. The study does not establish that condition in a customer population, a savings multiple, a market size or an optimal price.

7.3 B. Complete-cost benefit of the retained recipe

Over the observed 96-job episode, the recipe avoids 32 production labels relative to ordinary reuse and consumes 31 training labels, leaving one net label avoided. At two assumed minutes per reviewed example, that net label has $1.224333 opportunity value. Additional nonlabel recipe expense is $43.015834: a $0.03 session, 30 engineering minutes and one month's five-minute upkeep. The conditional net difference is approximately $-\$41.79$.

Against ordinary reuse, the tested episode would require 70.268 human minutes per equivalent example to repay this extra overhead under the base assumptions. Against active selection, production labels tie, so no finite positive labeling-time threshold repays the extra training and overhead. Tiny CPU differences do not change the whole-session allowance. The decision is **HOLD** for the learned recipe and retention of the active incumbent.

The retained recipe's amortized base cost of $1.249769 is below ordinary reuse's $1.628224, but that model extends the workload over future volume and useful life. It is not observed repayment of the 96-report episode. Active selection is less costly in all three modeled scenarios. This distinction prevents a favorable amortization assumption from replacing evidence of incremental benefit.

8 Financeable succession: affordability and payback windows

8.1 Nontransactional scenario design

The connected real-evidence ledger has unknown actual collections and full costs, so real financial succession stops. To examine the mechanism, we run a separately labeled 12-month simulation using the same native accounting contract. Six prespecified cases cross allocation ceilings of 0%, 25%, 50%, 75% and 100%, yielding 30 paths. No ratio is selected in advance as correct.

Common inputs include $5,000 external seed, $15 price per accepted report, 95% collection, 2% refunds, collection processing at 3% plus $0.30, a $500 reserve, $100 obligations and a 480-minute monthly construction budget. Base demand is 300 requests and expected operational acceptance is 98%, or 294 reports/month. The seed is not earned surplus. At this demand, operating expense is $382.62/month before improvement. Setup costs $1,425.96 once in cash; amortization is not also paid again.

The hypothetical next capability costs $336.69 plus $85.48 of failed construction, a complete $422.17. It consumes 387 assumed construction/review/example minutes. Its stipulated benefit is $0.50 per accepted reuse with $0.20 upkeep. With net benefit $0.30, complete fixed-cost repayment requires

$$N^* = \left\lceil \frac{422.17}{0.50 - 0.20} \right\rceil = 1{,}408 \text{ reuses.} \tag{7}$$

These are assumptions about a future capability, not the demonstrated benefit of the learned recipe. That recipe failed the strong-comparator gate and is not financed by this substitution.

Construction requires sufficient earned scenario surplus under the allocation ceiling, preserved reserves and obligations, remaining resource capacity, and enough remaining useful reuses for positive complete-cost repayment. Price, invoices, delayed collections, receivables, fees, costs and cash remain separate. The native ledger's observed-collected entries stay null; explicitly assumed scenario channels and adjacent projections retain the hypothetical receipts.

Table 7: All financing cases. Every cell is simulated. "No construction" can be a correct financial decision without proving a productive improvement.

Case	0% ceiling	25% ceiling	50–100% ceilings
Conditional success	No construction	Build month 3; repay month 10	Build month 2; repay month 9
Adverse demand	No construction	No construction	No construction
Failed capability	No construction	Lose $422.17; no benefit	Lose $422.17; no benefit
Zero saving	No construction	No construction	No construction
Protected balances	Startup refused	Startup refused	Startup refused
Delayed collections/payback	No construction	No construction	No construction

8.2 C. What the simulation establishes

In the conditional-success case, one-month collection delay and two-month benefit delay allow construction in month 2 at 50–100% ceilings, first benefit in month 5 and repayment in month 9. A 25% ceiling waits until month 3 and repays in month 10. Month 2 has $1,105.03 available earned scenario surplus; the minimum ceiling permitting this particular investment then is 38.21%. That number is a consequence of these inputs, not a reinvestment recommendation. Higher ceilings do not induce unnecessary additional construction.

The 50% success path ends with $42,124.56 cash and $4,410 receivables. Capability savings of $1,176, upkeep of $470.40 and the full $422.17 investment yield $283.43 net capability benefit by month 12. In the failure case the complete investment is lost, with no credit for benefits. The low-demand operation finishes with $2,017.74 of its seed and $147 receivable; remaining solvent over this horizon does not make it profitable.

Delayed collections expose a different failure. With four-month collection delay and three-month benefit delay, early periods lack earned available funds even when future value would suffice. By month 5 only 1,176 feasible reuses remain, below the 1,408 threshold. By the time surplus arrives, the value window has closed. In the protected-balance case, reserve $4,000 plus obligations $500 leave only $500 spendable, below setup cost; no operation starts. Zero saving never funds construction.

An independent Python rational-arithmetic audit passes 5,118 assertions across all 12 unit-cost rows, 30 financial paths and 360 monthly ledger snapshots. It checks cash and receivable conservation, no seed in earned surplus, protected balances, cost mapping, unknown propagation and investment gates. It does not validate the assumed demand, willingness to pay, future saving or operational reliability. Financeable succession is conditionally feasible in the stipulated-success world; actual financeability remains unknown.

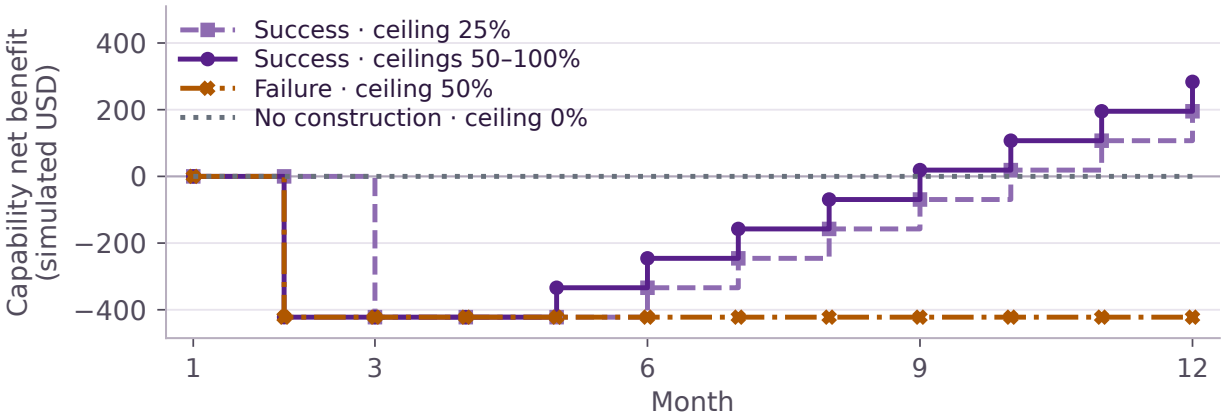
9 Ecosystem roles, authority and token boundaries

The inspected Alpha receipt path checks configured invoice-linked AGIALPHA transfers and records an earmark with `reinvestment_executed: false`. An earmark is not reconciled operating profit or an executed investment. Jobs v0's FeePool binds its configured AGIALPHA asset and supports accounted fees, configured burns and distribu-

Financeable construction depends on value and timing

SIMULATION · Hypothetical future capability · No observed receipts

A Benefits after complete construction cost and upkeep



$422.17 construction package, including failed work; $0.30 net benefit per reuse. Complete-cost payback requires 1,408 reuses. A funded failure earns no benefit.

B All 30 paths: year-end net benefit and construction → payback month

	0%	25%	50%	75%	100%
Conditional success	No buy	**+$195.23 M3 → M10**	**+$283.43 M2 → M9**	**+$283.43 M2 → M9**	**+$283.43 M2 → M9**
Low demand	No buy	No buy	No buy	No buy	No buy
Failed capability	No buy	**−$422.17 M3 → fail**	**−$422.17 M2 → fail**	**−$422.17 M2 → fail**	**−$422.17 M2 → fail**
Zero saving	No buy	No buy	No buy	No buy	No buy
Protected balances	No start	No start	No start	No start	No start
Delayed collections	No buy	No buy	No buy	No buy	No buy

Columns are allocation ceilings, not spending targets. Values are simulated USD; M denotes month. Low demand, zero saving, delayed collections and protected balances block construction.

Figure 4: Financing and useful-life windows. The same financial gate can fund a stipulated beneficial capability and reject late payback. All paths are simulations of a hypothetical future capability; they are not earned cash or demonstrated returns from the retained recipe.

Table 8: Actual contribution versus inspected receiving interfaces. Exact source identities are preserved in the companion.

Component	Contribution here	Boundary
Alpha / TI / SUCCESSOR	Existing Workbench extension, quote execution, plans, artifacts, identities and pack inspection.	New research adapter runs locally; integrity is supplemented by independent semantic verification.
SUCCESSOR Harness	Architecture and operational contract inspected.	Held proposals and authenticated execution not adopted or rerun.
Fleet Linux	Current route and delivery interfaces inspected.	No installed persistent worker, network recovery or production qualification executed.
AGI Jobs	Existing compiler emits seven unfunded package drafts through Workbench.	No registry submission, stake, funding, settlement or fee distribution.
AGI Job Manager	USDC settlement source inspected.	No Manager transaction or automatic conversion to AGIALPHA.
Shared ledger	Exact owner modules calculate costs and nontransactional decisions.	Modeled collections stay labeled; actual collections and surplus unknown.

tions to eligible stakers. Those conditional mechanisms do not establish universal passive-holder rights, enterprise-profit distributions or token appreciation [14].

Manager's inspected `JobSettlement.complete` route uses USDC accounting: specified recipient shares, a snapshotted validator budget and remaining agent reward, with failed transfers becoming claims. It does not automatically buy AGIALPHA. Fleet distinguishes `manager-usdc` from `agijobsv0-jobregistry-v2`; its inspected transfer record retains no authority, no production qualification, no independent review and settlement not connected. Deployed settings were not queried by RPC. We make no claim about live balances, live parameters, buybacks or financial returns [13, 12, 14].

Issue #4850 remains the coordination record. The new scope notice was posted there before this edition's publication work. Source-assurance and shared-cost ownership remain with the existing #4907 lane. The inspected #4899 runtime/planning lane still reports 56 unapproved security findings and an official scan exit of 1; review clearance is not security clearance. Original Harness and Fleet restrictions retain their original owners. No new agent, source revision, local demonstration or financial simulation resets those decisions. The prior local Chromium socket refusal was not retried or bypassed.

Publication uses the existing paper importer, catalog and generated page/service-worker pipeline, with a new immutable edition identifier. Software/demo artifacts and passive paper publication are distinct review lanes. A GitHub source file or draft PR is not a deployed public service. The companion records exact publication and link-check status separately from this manuscript's executed research inventory.

10 English/French demonstration and reproducible companion

The self-contained demonstration follows one continuous explanatory journey: mission, actual result, cost context, observed preparation limitation, executable construction, fresh qualification, admission decision and next mission. Its live run starts from the already established preparation boundary and constructs the adapter before the first new raw-input delivery; it does not rediscover that limitation on every click. The separate connected replay supplies the complete seed-to-next-job event chain. Three inspectable examples use the actual H01, H13 and H17 study inputs. They cover plain minor units, weekly lead times and currency affixes. Opaque headers are intentional experimental controls; the expanded input view explains them rather than implying real supplier data.

The English/French interface includes input inspection, the constructed JSON program, original report and CSV

downloads, a derived French report, evidence and financial-status views. The French presentation preserves actual numeric values and keeps the original English artifact and digest available. The deep-purple visual design, keyboard controls, reduced-motion behavior and responsive layout are implemented without network assets. The bundled runtime reuses the existing AWE module-bundling approach and exact Workbench modules; it does not create a second engine.

Local runtime checks inside the demo share prepared input and are labeled accordingly. The archived Python acceptance is a separate research result. If the user edits a mission, that archived independent acceptance does not transfer. No browser-executed independent oracle is claimed. Runtime/interface tests under Node exercise actual calculations and downloads; a browser visual run remains **NOT RUN** under the preserved session restriction.

The companion contains frozen source, data, all method outputs, failed/refused boundaries, construction traces, source licenses, cost and outcome ledgers, amendments, independent audits, the LLM action record, figure source and publication metadata. The core research requires Node and Python standard libraries; the static figure and paper rebuild additionally require the recorded plotting and TeX dependencies. A clean reproduction operates in a fresh output directory so original results are preserved.

```
# From the extracted companion directory:
python3 reproduce.py --output /absolute/path/to/new-reproduction

# Individual preserved-analysis entry points:
node implementation/controls.mjs
python3 economics/audit_results.py --results economics/results-v1 \
  --out /absolute/path/to/new-economic-audit.json

# Paper rebuild (XeLaTeX twice, from paper/):
xelatex -interaction=nonstopmode -halt-on-error AGI_ALPHA_Build_the_Next_Advantage.tex
xelatex -interaction=nonstopmode -halt-on-error AGI_ALPHA_Build_the_Next_Advantage.tex
```

11 Limitations and exact execution status

The experimental distribution is authored and finite. Its opaque headers isolate example-driven preparation but remove semantic header clues that a commercial LLM could exploit. The grammar is disclosed and narrow; all nominal tasks are in scope by design. Three held-out compositions test transfer within that grammar, not open-world schema induction, repository transfer or arbitrary multimodal work. Six families with four schemas each cannot establish real-world reliability or calibrated risk. Boundary refusal is valuable control behavior but cannot substitute for delivering harder useful work.

The independent oracle shares the research session and generated task specification, so common specification errors remain possible. We mitigate implementation error through separate language, arithmetic and search, mutation checks, source hashes and input-truth verification; these do not create an external replication. The LLM comparison is one actual constructor run, with standard tools, a reviewer and disclosed metadata exposure. It is neither a representative model population nor a matched-money test.

Acquisition count is an observed resource, but actual label-preparation effort is unknown. Human setup, engineering, model authoring, maintenance and customer effort are scenario values. The modeled whole-session charge is not a measured cloud tariff. Complete process CPU, observed customer payments and actual operating reserves are unknown. The economic model uses deterministic expected demand and fractional acceptance; it does not estimate stochastic ruin probabilities, contractual fixed commitments or demand response to price. Suspendible costs and assumed capability life materially affect outcomes.

The connected loop is a bounded research replay, not unattended production. The system constructs adapters with

researcher-provided grammar and tools; the broader agenda, mission family and resource model were selected by the research process. The learned recipe fails to beat conventional active selection. It would be misleading to infer self-improving intelligence, autonomous scientific discovery or sustained acceleration from the successful adapters.

Table 9: Execution inventory at manuscript freeze. Status is part of the result.

Activity	Status
Exact source and prior-edition inspection	EXECUTED
Frozen grammar, criteria, data and assumptions	EXECUTED
24 schema constructions per caching method	EXECUTED
96 fresh jobs per method; five methods	EXECUTED, all accepted
First-job qualification then subsequent reuse	EXECUTED
24 boundary probes, three selection policies	EXECUTED, all expected refusals
Current LLM-assisted constructor and replay	EXECUTED; billing UNKNOWN
Learned builder recipe transfer	EXECUTED; no strong-comparator gain
Complete-cost modeling / 30 finance paths	EXECUTED SIMULATION
Real collections / full actual cost / human duration	UNKNOWN
Real-money reinvestment / contract actions	NOT RUN
Authenticated Harness / persistent Fleet operation	NOT RUN
Browser visual execution / approved-profile install	NOT RUN
Economic promotion of learned recipe	HOLD
Sustained acceleration	NOT ESTABLISHED
Public website deployment of this edition	NOT CLAIMED

12 Conclusion

This edition advances AGI ALPHA from retaining an authored quote capability to deriving executable preparation programs from examples, qualifying them on fresh work and using them on subsequent missions through the existing runtime. It makes the full path inspectable, including the cost and authority boundaries.

The strongest result is specific: all five capable methods deliver the 96 fresh quote packages; active acquisition needs 29 examples instead of ordinary reuse's 61; the learned recipe provides no additional production benefit and fails complete-cost promotion. The useful input capability is retained, and the strongest conventional construction method remains in place.

Today's customer-value proposition does not depend on proving a self-financing Flywheel. The local service delivers complete exact comparison artifacts now. Human-work advantage remains conditional on measured-equivalent duration and operational assumptions. The financing mechanism identifies where a beneficial next capability could be affordable and where delayed collections, protected balances, low demand or failure should stop it. Actual cash-funded succession and improvement beyond the active incumbent remain research objectives, with the next experiment's decision criteria made explicit by this result.

A Source identities and preservation

The Alpha runtime is pinned to main `95d1e97d4d1c0aaebd592767fc5e5d2da56f6aa7`. The AWE companion is pinned to PR #4912 source `3d27c18362ed8db90dca4e6b85589865712bdbc5`; its inspected manuscript source is PR #4911 `0ae5e4b1abb627ed57ed261e16c95ed9d314d8a9`. The shared cost and ledger modules are taken from the #4907 owner source `2c5c393c1e5431c00d59d077c403739e789abd43`. Their earlier inspected owner revision had identical relevant module bytes.

Repository	Pinned inspected main
AGI-Fleet-Linux	45759fb096faf8988f58ae5ea71248d0f35a97c7
AGIJobsv0	bc4abbd5c28f3672275b13300e40ea26c820b8ee
AGIJobManager	5d5670df9dbbc79da115e0a44b71989f34df58c1

Identity	SHA-256
Primary freeze	d0d0f94598c86f3b76e22c78ea50944c58df115ad2639cdfe02c6d9388c9e9e7
Executable source record	45120cd168925ede3ab41f435cb87dfe736b89d1f75174fbaef714e0869b84a1
Dataset commitment file	845c88965caca5cf398e10e3e6e91cfab5da46f9eed42bbb9de06feead17a53c
Shared cost module	0dfca3ae4e7ef1d17bc78521f91917ef068280cf832d3565c354de2085c52506
Shared shadow ledger	cdc7c5a73593050a9af79b54589462e27f61b868e88ca6985d5cb639f3066d21
Frozen LLM programs	804fcb6d229c52b56f5123b4f7c3f55935b095129a59a80958ed112c13132d9e

The source manifest lists every vendored file and its original path, commit and hash. Older research remains at its original identifiers, including agi-alpha-work-economics-v1, agi-alpha-capability-construction-v1, agi-alpha-operational-economics-v1, agi-alpha-productive-capital-v1, agi-alpha-economic-engine-v1 and agi-alpha-autonomous-work-economy-v1. This edition uses agi-alpha-next-advantage-v1. It does not overwrite a prior paper, result archive or release asset.

B Failure, amendment and cost accounting record

The result archive retains all construction records, including eliminated candidate hypotheses. The nominal confirmation had no constructed-program rejection; boundary refusals remain separate. The learned recipe is a failed incremental-improvement hypothesis against the active comparator, not a discarded experiment. Failed future construction in the finance model is charged in full and remains explicitly hypothetical.

The preserved development runs show the evolution of generic safety and grammar checks before freeze. A double currency-affix rule was tightened prospectively. The LLM comparator corrected its own dollar-affix consistency rule before held-out acquisition; both source versions remain. These are engineering corrections, not post-result replacements of poor primary candidates.

Two reporting corrections were made after primary execution. The first restores the prospectively required family-stratified bootstrap after an initial pooled-schema reporting implementation; original output and code are retained in `evidence/reporting-v0`. The second adds packaging CPU to the auxiliary recorded-resource total while explicitly leaving complete CPU unknown. The frozen economic analyzer and original results remain intact. Its whole-session monetary allowance already covered packaging and verification process launch, so no cost tier or finance path changes. The companion names each additive amendment and binds its inputs.

No actual sales were available. The observed ledgers use null for collections, complete cost and available surplus. Simulated scenario receipts are carried through the figure data, native assumed-price channel and adjacent cash/receivable projections. A theoretical return, a customer saving and an external seed balance are never presented as earned operating proceeds.

C Research responsibilities and next decisive experiments

Vincent Boucher / MONTREAL.AI directs the research edition. AI-assisted work in the current session includes repository inspection, experimental design support, software and manuscript authoring, a separate data/oracle role, a separate LLM comparator, financial audit and demonstration work. All inspectable source and actions are retained; the process is not external peer review. Existing integration, security and publisher owners retain their original authority. Publication review may correct the edition additively; the frozen primary evidence must remain available.

The next useful experiments follow the observed limits. First, measure complete supervised service delivery on actual customer quote packages with fixed acceptance and auditable preparation time. Second, vary new-schema arrival, reuse and drift to test whether label savings survive realistic distributions. Third, evaluate any new builder capability against the active specialist and the same-resource current LLM constructor, with model billing and authoring effort observed. Fourth, admit a beneficial candidate only after those costs and sufficient useful life are demonstrated. Finally, test repeated succession only when multiple such admitted cycles actually exist. These are future tests, not executed outcomes in this edition.

References

[1] S. Gulwani. *Automating String Processing in Spreadsheets Using Input–Output Examples*. POPL, 2011. Primary research page: https://www.microsoft.com/en-us/research/publication/automating-string-processing-spreadsheets-using-input-output-examples/.

[2] MONTREAL.AI. *AGI-Alpha-Agent-v0*, source `95d1e97d4d1c0aaebd592767fc5e5d2da56f6aa7`. Workbench, TRANSFORMATIONAL INTELLIGENCE guide and SUCCESSOR interfaces. https://github.com/MontrealAI/AGI-Alpha-Agent-v0/tree/95d1e97d4d1c0aaebd592767fc5e5d2da56f6aa7.

[3] V. Boucher / MONTREAL.AI. *AGI ALPHA FLYWHEEL: The Autonomous Work Economy*. AWE-2026-01. Immutable manuscript PR 4911 and companion PR 4912, inspected 25 October 2026. https://github.com/MontrealAI/AGI-Alpha-Agent-v0/pull/4911; https://github.com/MontrealAI/AGI-Alpha-Agent-v0/tree/3d27c18362ed8db90dca4e6b85589865712bdbc5/studies/agi-alpha-autonomous-work-economy-v1.

[4] V. Boucher / MONTREAL.AI. *Economics of Autonomous Work*. Immutable paper `agi-alpha-work-economics-v1`, merged paper PR 4902. https://github.com/MontrealAI/AGI-Alpha-Agent-v0/pull/4902.

[5] V. Boucher / MONTREAL.AI. *From Useful Work to Reusable Productive Capability*; *Complete Economics of Accepted Work*. Immutable papers `agi-alpha-capability-construction-v1` and `agi-alpha-operational-economics-v1`. Source `5caa871db78352557699f783027cf64357403d8b`. https://github.com/MontrealAI/AGI-Alpha-Agent-v0/pull/4903.

[6] V. Boucher / MONTREAL.AI. *The Productive Capital Engine*. Immutable paper `agi-alpha-productive-capital-v1`; source/demo PR 4909. https://github.com/MontrealAI/AGI-Alpha-Agent-v0/pull/4908; https://github.com/MontrealAI/AGI-Alpha-Agent-v0/pull/4909.

[7] V. Boucher / MONTREAL.AI. *The Economic Engine*. Immutable paper `agi-alpha-economic-engine-v1`; source `67e664f419511c59f41f426171614d7f425ed67f`. https://github.com/MontrealAI/AGI-Alpha-Agent-v0/pull/4910.

[8] MONTREAL.AI. *Automated costs and shadow ledger specifications*. Source-assurance owner PR 4907, inspected source `2c5c393c1e5431c00d59d077c403739e789abd43`. https://github.com/MontrealAI/AGI-Alpha-Agent-v0/pull/4907.

[9] U.S. Bureau of Labor Statistics. *Employer Costs for Employee Compensation—June 2026*. Archived 9 September 2026 release, private-industry occupational groups. Accessed 25 October 2026. https://www.bls.gov/news.release/archives/ecec_09092026.htm.

[10] U.S. Bureau of Labor Statistics. *May 2025 National Occupational Employment and Wage Estimates*, procurement clerks, occupation 43-3061. Accessed 25 October 2026. https://www.bls.gov/news.release/ocwage.t01.htm.

[11] MONTREAL.AI. *AGI ALPHA coordination record*, issue 4850. Study scope notice: https://github.com/MontrealAI/AGI-Alpha-Agent-v0/issues/4850#issuecomment-6403108438. Existing owner action map: https://github.com/MontrealAI/AGI-Alpha-Agent-v0/issues/4850#issuecomment-6386746278.

[12] MONTREAL.AI. *AGI-Fleet-Linux*, source `45759fb096faf8988f58ae5ea71248d0f35a97c7`, including `local-runtime/project/transfer.mjs`. https://github.com/MontrealAI/AGI-Fleet-Linux/tree/45759fb096faf8988f58ae5ea71248d0f35a97c7.

[13] MONTREAL.AI. *AGIJobManager*, source `5d5670df9dbbc79da115e0a44b71989f34df58c1`, `contracts/utils/JobSettlement.sol`. https://github.com/MontrealAI/AGIJobManager/blob/5d5670df9dbbc79da115e0a44b71989f34df58c1/contracts/utils/JobSettlement.sol.

[14] MONTREAL.AI. *AGIJobsv0 FeePool*, source `bc4abbd5c28f3672275b13300e40ea26c820b8ee`; Alpha receipt path, source `95d1e97d4d1c0aaebd592767fc5e5d2da56f6aa7`. https://github.com/MontrealAI/AGIJobsv0/blob/bc4abbd5c28f3672275b13300e40ea26c820b8ee/contracts/v2/FeePool.sol; https://github.com/MontrealAI/AGI-Alpha-Agent-v0/blob/95d1e97d4d1c0aaebd592767fc5e5d2da56f6aa7/alpha_factory_v1/core/runtime/chain.py.