

TECHNICAL PAPER / SIMULATION AND METHODS

Regional capacity and managed inference cost floors

DIMAGGI AI
4 October 2026

Abstract

An inference service needs a compatible workload, eligible locations, capacity available at peak demand, recovery inventory and a reconciled cost envelope. We examine the existing offline regional placement, pricing and paired-offer models using their frozen synthetic examples. The regional screen rejects a policy-ineligible pool even when it has spare capacity. The paired offer costs two whole accelerator units at each of the primary and standby pools. A separate capped-use example produces a \$4,717.68 stress cost and \$0.006419 request floor, but requires 749,955 billable requests to recover that period cost under its credit assumption. These are conditional arithmetic findings, not observed customer economics. Public aggregate inference observations do not supply the phase-specific calibration needed by the price model. No result establishes legal compliance, available supplier capacity, profitable operation or an SLA.

Evidence class	Deterministic synthetic pricing and placement; selected published benchmark observations retain their scope. No local hardware benchmark.
Exact study source	5c4c8434dd0f92556aed46b4ebcda0c11ae468de
Reproducibility	Frozen inputs, result hashes, figure data and implementation test receipts accompany the public paper.
Interpretation	Model QA does not establish independent technical acceptance, a production SLA, legal compliance or realized commercial outcomes.

Full text, accessible figures and evidence: dimaggi.ai/research/regional-inference

Research question and contribution

The question is whether the existing planning interfaces retain the distinctions between policy eligibility, whole-replica peak reserve, period costs and request billing when composing a proposed regional service. This is a deterministic interface and accounting study, not a vendor comparison or optimal-placement study. It complements the [capacity and lifecycle paper](#) and reuses the existing network, scheduling, cooling and reliability owners. No owner model is reimplemented here.

The hypotheses are deliberately narrow: spare pool capacity cannot override declared locality rules; average period utilization cannot replace a peak reserve; and a feasible arithmetic quote cannot prove either demand or operational qualification. Refusal cases and the persistence of cost at low sales are part of the result, rather than discarded exceptions.

Methods and reproducibility

The [reproduction appendix](#) records the source pin, environment, commands and artifact hashes. The wrapper runs the three existing CLI examples and the existing public-data analysis without changing their inputs or engineering code. The [manifest](#) checks every exported model, fixture and test file against the candidate; private coordination documents are excluded from the public bundle. There is no stochastic sampling or seed. Results are exact under the declared inputs; there is no empirical population from which to infer confidence intervals. The existing 186-test suite and Go race tests were rerun; their success establishes tested implementation behavior only.

Fixture	Purpose and unit	Material assumptions
placement-example.json	Screen one added workload; requests/s per complete replica	Synthetic DE/US pools, 80 peak requests/s, 100 good requests/s per replica, 0.7 utilization ceiling; supplied latency, age and failure-domain labels
pricing-example.json	Capped-use period cost; USD per bounded request	1,000,000 planned requests over 720 hours; 2,048 input and at most 512 output tokens; at most two attempts; synthetic throughput and rates; one primary plus one failover accelerator
offer-example.json	Bind peak reserve to a predictable quote	Different synthetic phase throughput and minimum allocation; one accelerator per whole replica, two allocated at each pool
public-data provenance	Parse pinned third-party observations with source and unit boundaries	Public benchmark observations remain whole-system, scenario-specific observations; geography and operational sensitivity inputs remain synthetic

The standalone price fixture and paired-offer fixture differ in performance inputs and service class. They are not a controlled comparison of reservation policies. In particular, \$4,717.68 is not the cost of the paired 2+2 offer.

For peak demand P , resident demand R , declared replica goodput g and utilization ceiling u , the placement screen requires $\text{ceil}((P + R)/(g \cdot u))$ total replicas. The paired offer separately reserves whole resident replicas and costs the incoming workload's dedicated reserve. It uses the larger of the goodput-based incoming reserve and a serialized uncached prefill/decode demand bound, multiplied by the maximum attempt count. Standby must carry the full incoming peak after resident reservation. Rounding occurs at whole-replica boundaries. Supplied independence labels must differ in each required dimension, but labels do not measure correlated failure probability.

The price model adds prefill and decode seconds per attempt, then computes $\text{busy_hours} = \text{requests} * \text{attempts} * \text{service_seconds} / 3600$. Primary quantity is at least the declared minimum and otherwise the ceiling of busy hours divided by $\text{period_hours} * \text{utilization} * (1 - \text{headroom})$. Failover quantity is a ceiling of the chosen fraction of primary quantity. The expected case uses declared cache hits and expected extra attempts. Both cases use maximum output tokens; stress uses cache misses and maximum attempts. These serialized, linear demand assumptions do not predict batch overlap, memory fit, queueing, time to first token or tail latency.

Cost includes all provisioned primary and failover hours, disjoint compute, energy/cooling and interconnect components, storage, network usage, platform, observability, support, licensing and the stated risk reserve. USD has no inferred FX or tax conversion; a GB is 10^9 bytes. The 720-hour price period is distinct from the lifecycle model's actual calendar-month hours. Already included energy or platform components must be zeroed with supporting attribution, rather than charged twice. Integer ceilings use rational arithmetic; money uses the model's explicit Decimal context. Unit prices round upward to microdollars and invoices upward to cents.

Findings

RI 01 Regional eligibility remains a policy screen

The [placement result](#) retains the two DE-labelled candidates and rejects `example-us-c` for all eight declared country planes: processing, storage, backup, telemetry, control, keys, operators and transit. The rejected pool has a modeled 700 requests/s of warm spillover, so unused modeled capacity does not override those inputs. The result says `candidate_for_review`, with `production_qualified` and `execution_authority` false. This is an application of caller policy to caller-supplied data, not a legal conclusion or authenticated regional inventory. Google Cloud's resource location documentation also distinguishes creation restrictions from data storage commitments; it is related scope guidance, not validation of this model. [Google Cloud](#)

RI 02 Whole peak reserves remain in the paired quote

The [paired-offer result](#) requires two accelerator units at the primary pool and two at standby. Available capacity after resident reservation is also two at each. Its stress cost is \$8,756.88, fixed unit floor \$0.011915 and predictable planned commitment \$11,915.00. These numbers describe the distinct paired synthetic fixture. A real eight-accelerator replica would require whole groups of eight and matching calibration; the example does not establish that one device can serve an actual workload. The adapter does not allocate live capacity, and `tool_guard_evaluated` is false.

A regional quote preserves separate policy and capacity checks

Paired-offer fixture only: caller-supplied locations, independence labels, throughput and rates.



Declared country planes

Processing, storage, backup, telemetry, control, keys, operators and transit.

Offline candidate only. No live reserve, legal determination, SLA or execution authority.

Synthetic planning results | Source: Decision Application 5c4c843 | 4 October 2026

Figure 1. The paired-offer fixture keeps declared policy eligibility, peak reserve, period cost and customer quote as separate checks. The standby allocation can carry the incoming peak under supplied linear capacity assumptions. Differing labels are not evidence of real failure independence or lawful processing. [Figure data \(CSV\)](#).

RI 03 A cost floor retains demand risk

The existing pricing interface proposes predictable commitment, capped flexible usage and interruptible usage, each with a fixed customer rate for a bounded envelope. Payment class does not itself qualify recovery or availability; those remain separate capacity and evidence requirements. These are proposed classes, not accepted products.

The separate capped-use result has stress cost \$4,717.68, unit floor \$0.006419 and a \$6,419.00 planned invoice at one million requests. Its full period cost divided by the per-request rate after the 2% reserved credit fraction requires 749,955 billable requests, rounded up. That is a conditional cost-recovery threshold, not a sales forecast. Fixed provisioned capacity remains costly if fewer requests arrive; the planned allowance and maximum affordable requests do not establish demand. Maximum contractual credits in this fixture equal the whole planned invoice, leaving zero net revenue in that extreme. The expected credit reserve does not fund that maximum liability.

A price floor still needs enough billable requests

Standalone capped-use fixture only: one primary plus one standby accelerator; one 720-hour period.



$\text{ceil}[4,717.68 \text{ USD} / (0.006419 \text{ USD/request} \times 0.98)] = 749,955$

This calculation is not a demand forecast. Maximum contractual credits can reduce net revenue to zero.

Synthetic planning results | Source: Decision Application 5c4c843 | 4 October 2026

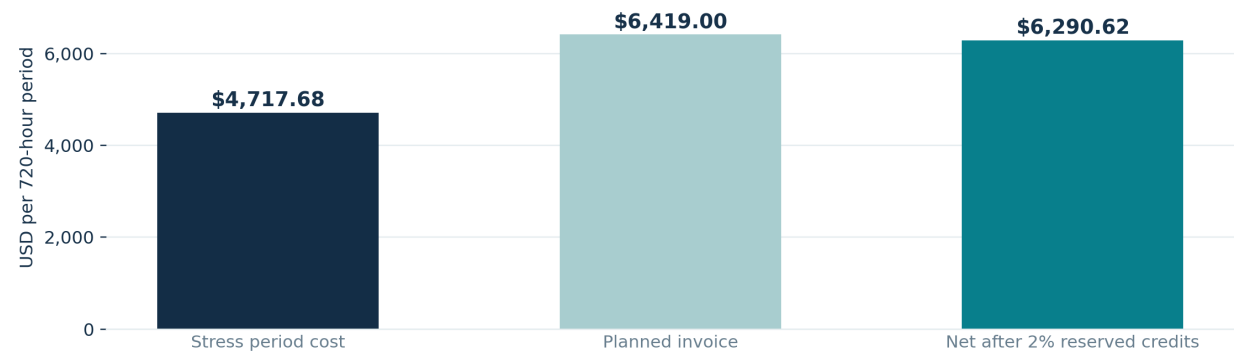
Figure 2. The threshold retains the full stress period cost and assumes the fixed bounded-request rate with 2% reserved credits. A variable-cost volume experiment would be required for a broader profitability curve. The maximum contractual credit exposure is larger than the expected reserve. [Figure data \(CSV\)](#).

RI 04 Margin and markup have different denominators

For cost c already including the risk reserve, target net gross margin m and reserved credit fraction c , the pre-rounding invoice floor is $C / ((1 - c) * (1 - m))$. Realized modeled margin is $(\text{net_revenue} - C) / \text{net_revenue}$; invoice markup is $(\text{invoice} - C) / C$. The capped-use fixture reports about 25.005% net gross margin using \$6,290.62 revenue after reserved credits as its denominator, and about 36.063% invoice markup using \$4,717.68 cost as its denominator. At the maximum contractual credit exposure, net revenue is zero and margin is undefined; it is not a finite negative percentage. Both reported percentages describe the same modeled planned volume and reserved-credit assumption. Neither measures actual profit, demand, credit eligibility or an accepted price. See the exact Decimal outputs in the [price artifact](#).

A price floor separates cost, invoice and credit exposure

Standalone capped-use fixture only: 1 primary + 1 standby; 1,000,000 bounded requests; 720 hours.



Net margin: $(6,290.62 - 4,717.68) / 6,290.62 = 25.005\%$

Invoice markup: $(6,419.00 - 4,717.68) / 4,717.68 = 36.063\%$

The paired 2+2 offer is a separate fixture and is not compared here.

Synthetic planning results | Source: Decision Application 5c4c843 | 4 October 2026

Figure 3. The standalone cost floor retains provisioned primary and standby capacity over the full 720-hour period. The margin denominator is net revenue after reserved credits; the markup denominator is cost. These are planned-volume calculations using synthetic rates, not observed profit or market prices. The 2+2 offer is a distinct fixture documented separately. [Figure data \(CSV\)](#).

RI 05 The local budget invariant has an operating boundary

The existing BudgetBook requires spent plus held money to remain within its cap and refuses settlement above a reservation. Tests exercise concurrent threads, replay and excessive settlement. It is process-local, loses state on restart and does not cancel supplier work. This supports a prototype invariant, not a durable production spending cap. A multi-worker admission ledger, bounded in-flight liability and billing reconciliation require additional implementation and evidence. AWS documents that budget notifications can lag incurred charges; this motivates the distinction between monitoring and admission control without establishing a guarantee for either system. [Source and tests, AWS Budgets](#)

RI 06 Aggregate public observations cannot fill missing phase calibration

The [frozen refusal record](#) shows that pricing_calibration refuses both selected AMD MI355X and NVIDIA B300 inference records because they lack separate prefill, cached-prefill and decode measurements. The [public-data result](#) preserves original observation scope and synthetic placement sensitivity labels. No GPU was run locally. Whole-system benchmark throughput cannot be divided into per-device phase rates without a justified scaling model. Hardware specification fixtures in the lifecycle paper are a further, separate evidence class. MLPerf's workload, quality and scenario conditions inform future measurement design; parsing its results does not qualify this application or its prices. [MLPerf Inference documentation](#)

Lifecycle and governance interfaces

The existing lifecycle bridge permits a fully attributed, committed bundle with matched hardware, profile, calendar and calibration identity. It distributes bundle cost over primary device-hours while retaining the paid recovery scope; it refuses duplicate cost components and unsupported workload conversions. This connects accounting interfaces, not verified hardware performance. The separate lifecycle orchestrator invokes the real Tool Guard v0.3.0 evaluator for offline policy decisions. Allowed planning and escalated procurement/network/ retirement proposals do not authorize effects. Execution and verification remain outside this planning model. A hash binds content; it does not authenticate its issuer or record a review. See [contracts](#) in the public source bundle.

Additional experiments required

Question not answered here	Necessary design before a stronger claim
What service can one actual replica sustain?	Matched model, precision, prompt/output distribution, cache state and quality; measured phase rates, memory fit and client-visible TTFT/TPOT under stepped load; repetitions and uncertainty
Does regional recovery preserve the service?	Authorized site evidence and correlated dependency failures; resident peak load, routing/state restoration and failback; good/total request denominator including admitted failures and specified cap refusals
Does a cap survive operational faults?	Durable concurrent reservation, crash/restart, duplicated and delayed receipts, supplier cancellation failure, bounded retry/fan-out and reconciliation against final invoices
Can the proposed price recover costs?	Dated supplier contracts with units and inclusion rules; measured operations/support cost, demand distribution, credits and sensitivity to unsold reserve; separate commercial approval
Do locality inputs correspond to actual processing?	Workload-specific approved policy plus authenticated storage/access/transit and subprocessor evidence; separate legal assessment, not an experiment that can prove lawfulness

An availability SLI should state good events, total eligible events, window and exclusions before a target is considered. Google SRE provides a useful method for that definition; this study contains no production request population. [Implementing SLOs](#)

Evidence checklist

Findings	Public evidence	Interpretation check
RI-01	Regional placement result	Policy is supplied; location and legal status are not attested.
RI-02	Paired-offer result	Distinct 2+2 fixture; no comparison of policy benefits against the 1+1 fixture.
RI-03, RI-04	Standalone price result	Preserve 720 hours, one million planned bounded requests, credit reserve and maximum liability.
RI-05	Byte-exact source tests and reproduction instructions	Process-local budget invariant; no durable production spending cap.

Findings	Public evidence	Interpretation check
RI-06	Calibration refusals and public-data analysis	Whole-system observations cannot fill absent phase calibration.

Reproducibility appendix

The [public evidence bundle](#) contains the permitted byte-exact model source, tests, fixtures, frozen outputs and a packaging-only reproduction wrapper. The [manifest](#) binds the exported files to source `5c4c8434dd0f92556aed46b4ebcda0c11ae468de`. [Reproduction instructions](#) describe the environment, source verification, model execution and test commands. The cooling owner is pinned to `735d8eb8196b70aa6beea759bc3c9470ce286a74`. No hardware or cloud workload is executed.

The [clean-extraction validation receipt](#) records 186 passing Python tests and successful Go race checks. The separate [reproduction receipt](#) records equality for all nine scenarios, the 60-month plan, four CLI outputs and two calibration refusals. These receipts are implementation QA.

Unpack the archive into a new folder, read its README, and first run `python3 reproduce.py --verify-only`. Full replay requires Python 3.12, Go 1.26.1 and the named cooling-owner checkout; use the exact commands in the bundle. The original scenario outputs and their negative results remain unchanged. Variable elapsed-time metadata is excluded from numerical equality; it is not a new performance benchmark.

Source links in the body that target the archive refer to the path named in the link or adjoining text. Key files include `service_planning/lifecycle/experiments.py`, `service_planning/lifecycle/docs/CONTRACT.md`, `service_planning/lifecycle/docs/SOURCES.md`, the JSON examples, the pricing/placement/offer modules and their tests. The public bundle omits private coordination documents and does not expose the owning repository's full history.

References

1. [Google Cloud: resource-location restrictions](#). Location-policy scope does not establish lawful processing.
2. [AWS Budgets: managing costs](#). Budget notifications and admission control have different operating boundaries.
3. [MLPerf Inference documentation](#). Workload, quality and scenario conditions must accompany observations.
4. [Google SRE Workbook: implementing SLOs](#). Define good events, eligible events, window and exclusions before applying a service objective.