

PECA v2: Anticipatory Execution for Incremental AI Agents: A Position Paper on Multi-Future Forecasting, Reversible Speculation, and Latency Hiding

RM369

Abstract

Prefix-Conditioned Execution and Cumulative Audit (PECA) v1 treats incremental interpretation and cumulative verification as parts of an agent’s execution controller. This paper proposes PECA v2, which adds an anticipatory layer: while instructions are being received or a task is taking shape, the agent forecasts likely future requirements and prepares low-cost work before those requirements are confirmed. Forecasts are separated into input, task, and preference channels; each carries evidence, calibrated confidence, a time horizon, a confirmation or contradiction condition, and a suggested verifier. Rather than betting everything on one predicted future, the controller should prefer preparation that remains useful across several plausible continuations, reserving branch-specific work for isolated and cheaply reversible experiments. The proposal may hide latency and improve collaboration in streaming or long-running tasks, but it can also introduce false constraints, over-personalization, wasted computation, and self-reinforcing assumptions. In this version, high-confidence predicted requirements enter a tagged cumulative review alongside confirmed requirements; predicted-only failures may be repaired speculatively or demoted rather than treated as mandatory failures. This keeps forecasts visible but risks letting the agent’s own expectations influence the apparent definition of success, a limitation addressed by a stricter successor. PECA v2 is a design hypothesis, not a measured performance improvement.

1. Motivation

An effective collaborator often prepares before a request is fully spelled out. A developer describing a feature may eventually need tests, a migration plan, or an installation note. A user dictating a task may pause between clauses while an agent could already inspect the relevant workspace. The predictive opportunity is not unrestricted mind-reading: it is a forecast conditioned on the observed instruction prefix, the current task state, and relevant authorized project history.

PECA v1 makes the observed prefix operational [9]. It updates a traceable requirement ledger, changes a persistent artifact at meaningful semantic boundaries, and audits the work against the requirements understood so far. That loop addresses a reactive failure mode: a constraint can be omitted while the artifact is being built and only discovered during a final review. V2 asks a different question: can an agent safely prepare for likely future needs before the user explicitly asks for them?

Consider a user who says, “Help me add search to this small research app.” The agent might inspect the current data model and prepare a disposable test outline while it waits for more details. It should not automatically build a full-text indexing service, because the user may next say “keep it minimal” or “do not change the schema.” The useful capability is anticipation with asymmetric commitment: preparation can start early, but an assumption must not silently become a requirement.

This paper defines a controller-level proposal and falsifiable hypotheses. It does not claim that prediction is inherently accurate, that a model’s reported confidence is calibrated, or that anticipation always saves time.

2. Relation to prior work

PECA v2 builds on, but does not replace, work on agent action loops, self-refinement, speculative execution, personalization, and confidence estimation.

Agent feedback loops. ReAct interleaves reasoning with actions and environmental observations [1]. Self-Refine iteratively critiques and revises a candidate output [2], while Reflexion uses verbal feedback and episodic memory to influence later attempts [3]. PECA’s anticipatory extension focuses on forecasting requirements and useful preparation while retaining a persistent artifact and a source-linked requirement ledger.

Speculative execution. PASTE studies pattern-aware speculative tool execution to overlap likely tool activity with model generation [4]. That work is directly relevant to latency hiding, but tool-call prediction is not identical to predicting an unstated user requirement. PECA v2 treats both as possible forecast channels and requires any branch-specific preparation to be bounded by reversibility, value, and authorization.

Preference and confidence limits. PrefEval finds that inferring and following preferences across

long conversations remains challenging for current LLMs [5]. A survey of confidence estimation and calibration likewise documents substantial methodological challenges in turning model confidence into trustworthy probability estimates [6]. V2 therefore treats confidence as a quantity to calibrate against outcomes, not as a self-authenticating claim.

Verified speculative work and audit budgets. Certificate-Gated Prefix Acceptance proposes trusted verification to gate speculative prefixes under hard constraints [7]. BRA-Audit studies budget-aware placement of runtime audits in multi-agent trajectories [8]. These works are important neighboring efforts. PECA v2’s proposed scope is anticipation conditioned on instruction evolution and task context, coupled to a predictive requirement ledger and reversible preparatory actions. Whether this combination provides a useful incremental contribution remains an empirical question.

3. Architecture

The state of the controller is represented as

$$\Sigma = (R_h, R_p, S),$$

where R_h is the ledger of active, confirmed requirements; R_p is the predictive ledger; and S is the persistent working artifact together with its audit evidence and change history. Input buffers, counters, and scheduling metadata may be derived from an event log rather than treated as additional sources of authority.

3.1. Three prediction channels

The anticipator emits a small set of plausible requirements or useful next actions, preferably across more than one plausible future.

1. **Input anticipation** forecasts what an unfinished message or streamed instruction may say next. Its horizon is short and its evidence should come from the observed prefix, not from hidden future text.
2. **Task anticipation** forecasts supporting work likely to be useful for the task already understood, such as tests, dependency inspection, or a rollback plan.
3. **Preference anticipation** forecasts presentation or workflow preferences from relevant, authorized project or user history. It must be scoped to the current context and should decay when evidence is stale or contradictory.

Every predictive entry includes its claim, channel, source evidence, confidence, scope, expiry condition, confirmation signal, contradiction signal, suggested verifier, and estimated cost of discard. A rationale is not itself evidence: “the user often asks for tests” should point to applicable observed instances or remain explicitly uncertain.

3.2. The six-phase micro-cycle

Each meaningful semantic unit (not necessarily each character or token) runs through six phases:

1. **Sense.** Accept newly observed text, tool results, or material task-state changes.
2. **Interpret.** Reconcile the observed prefix into R_h , preserving source spans and superseded or unresolved constraints.
3. **Anticipate.** Update R_p from the prefix, task state, and authorized relevant context; keep multiple plausible continuations when uncertainty matters.
4. **Select.** Prefer robust preparation useful across several futures. Branch-specific speculation is eligible only when its expected value is positive, confidence is sufficiently calibrated, and its effects are cheaply reversible.
5. **Act.** Execute confirmed work at its normal authorized commitment level; isolate speculative changes in a draft, sandbox, or versioned branch.
6. **Audit and update.** Review the artifact against active requirements in R_h together with high-confidence entries from R_p , but tag every finding by origin. A hard-requirement failure requires repair or explicit unresolved status. A predicted-only failure may lead to optional reversible preparation or to reducing the forecast’s confidence; it must not be promoted into R_h without supporting input or other authorized evidence. Reconcile new evidence, roll back contradicted dependent work, and checkpoint the state.

3.3. Selection by expected value

A forecast is not valuable merely because it is plausible or later proves correct. A simple action score is

$$U(a) = V_s - C_e - C_d - C_i - R_r,$$

where V_s is expected saved work, C_e is execution cost, C_d is expected discard cost, C_i is integration cost, and R_r is residual risk. This is a decision aid, not a calibrated guarantee; estimates must be tested. If no safe action has positive expected value, anticipation should produce no action.

Robust preparation is preferred: inspect a project, prepare a test harness, or map dependencies when that work supports several plausible next steps. Branch-specific speculation can explore one forecast, but it must be cheaper to discard than to keep. External side effects such as sending messages, purchasing services, deleting data, changing permissions, or deploying code require the relevant authorization and a separate commit gate regardless of predictive confidence.

3.4. Nested review rhythm

V2 proposes a six-phase micro-cycle, a meso-review after each three completed cycles, and a strategic review after each nine cycles. The meso-review recalibrates predictions and measures precision, recall, and speculative waste. The macro-review examines stale assumptions, trajectory coherence, and whether the prediction strategy should be reduced or disabled. The 3-6-9 schedule is a hypothesis to ablate, not a claim that those numbers possess a special technical property. Material safety events and contradictions must trigger immediate review rather than waiting for a counter.

4. Evaluation plan

The principal hypotheses are: (H1) anticipation reduces end-to-end latency when input is streamed or task work has predictable support actions; (H2) multi-future robust preparation retains more useful work than a top-one forecast; (H3) scoped project history helps established tasks but can hurt cold-start or shifting contexts; and (H4) periodic recalibration improves net utility only when it offsets its cost.

A controlled study should compare the same base model, tools, sampling settings, prompt set, and budget across direct execution, PECA v1, full PECA v2, v2 with predictions computed but not acted upon, top-one versus multi-future anticipation, no-history versus authorized project-history anticipation, and the full system with fixed 3-6-9 versus event-triggered review.

Use streaming and completed-prompt tracks separately. In the streaming track, the agent must not receive future text before it arrives. In the completed-prompt track, anticipation concerns next actions and supporting work rather than a hidden remainder of the prompt. Include long collaborative sessions, cold-start users, stale preferences, abrupt preference changes, late negations, minimality constraints, and prompts where the most likely predicted feature is explicitly unwanted.

Primary metrics should include explicit-requirement satisfaction and critical violations. Anticipation-specific metrics should include forecast precision and recall, calibration error or Brier score, retained speculative work, discarded-work fraction, net latency saved, action reversals, user correction rate, and total tokens or tool cost. Evaluate with deterministic checks where possible and blind human review for subjective outputs. A model grading its own forecasts should not be the sole source of labels.

5. Failure modes and safeguards

False prediction as hidden obligation. V2's initial cumulative review considers the union of confirmed requirements and high-confidence predictions, with findings tagged by origin. Even when predicted-only findings are not mandatory, putting them in the same review operation creates pressure to satisfy expectations the user never expressed. This can expand a deliberately minimal task and make the agent optimize a self-generated proxy for success. Hard-compliance scores and forecast utility must therefore be reported separately. A stricter successor should place the two audits in separate runtime paths so predictions cannot affect whether the artifact passes the hard-requirement test.

Self-reinforcing personalization. If the agent prepares extra features and users tolerate them, the system might incorrectly conclude that every user wants elaborate outputs. Acceptance, explicit confirmation, and explicit rejection are different signals. Preferences should be scoped, age-weighted, and easy to reset.

Confidence without calibration. A number emitted by a model is not a probability guarantee. Calibrate by prediction type and horizon; report calibration under distribution shift; disable speculative action when evidence is thin.

Premature or costly preparation. Early action can encode a wrong interpretation or create indirect dependencies. Use isolated branches, dependency graphs, checkpoints, and rollback tests. Reversibility must include downstream effects, not only the direct edit.

Privacy and authorization. Preference models should use only context the system is authorized to use. Inferred intent never authorizes external side effects. Forecasting must not disclose or unnecessarily retain sensitive history.

6. Limitations and conclusion

PECA v2 is a controller proposal, not a trained model or an evaluated product. No empirical results are reported here. The anticipated benefits depend on forecast calibration, useful lead time, cheap reversibility, and tasks where preparation can happen in parallel with instruction arrival. In one-shot prompts with little predictable follow-up, prediction may simply add cost. In variable or cold-start contexts, preference anticipation may be harmful. The architecture also faces significant overlap with the established fields of speculative tool execution, personalization, and agent verification; a distinct contribution must be demonstrated rather than assumed.

PECA v2 extends incremental execution with a forecast-driven preparation path. Its most valuable

principle is asymmetric commitment: act early only in ways that are safe to undo, and use new evidence to confirm, revise, or discard the preparation. The decisive next step is not to maximize anticipation, but to measure whether the saved time and retained work exceed computation, discard, integration, and risk costs. A follow-on version should enforce a strict separation between predicted needs and the criteria that determine whether the user’s actual request has been satisfied.

Drafting and reproducibility note

This position paper was prepared with language-model assistance. It presents a proposed architecture and an evaluation plan, not completed empirical research. The author should verify citations, conduct a broader prior-art review, revise the manuscript in their own voice, and follow the repository’s disclosure policies before publication.

References

- [1] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. ReAct: Synergizing Reasoning and Acting in Language Models. arXiv:2210.03629. <https://arxiv.org/abs/2210.03629>
- [2] Aman Madaan et al. 2023. Self-Refine: Iterative Refinement with Self-Feedback. arXiv:2303.17651. <https://arxiv.org/abs/2303.17651>
- [3] Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language Agents with Verbal Reinforcement Learning. In *Advances in Neural Information Processing Systems 36*. <https://arxiv.org/abs/2303.11366>
- [4] Yifan Sui, Han Zhao, Rui Ma, Zhiyuan He, Hao Wang, Jianxun Li, and Yuqing Yang. 2026. Act While Thinking: Accelerating LLM Agents via Pattern-Aware Speculative Tool Execution. arXiv:2603.18897. <https://arxiv.org/abs/2603.18897>
- [5] Siyan Zhao, Mingyi Hong, Yang Liu, Devamanyu Hazarika, and Kaixiang Lin. 2025. Do LLMs Recognize Your Preferences? Evaluating Personalized Preference Following in LLMs. In *International Conference on Learning Representations (ICLR)*. https://proceedings.iclr.cc/paper_files/paper/2025/hash/28a46044775d97a4efcbcf14e7f13209-Abstract-Conference.html
- [6] Jiahui Geng, Fengyu Cai, Yuxia Wang, Heinz Koeppl, Preslav Nakov, and Iryna Gurevych. 2024. A Survey of Confidence Estimation and Calibration in Large Language Models. In *Proceedings of NAACL-HLT 2024*, 6577–6595. <https://aclanthology.org/2024.naacl-long.366/>
- [7] Chenyu Zhou, Qiliang Jiang, Shuning Wu, and Xu Zhou. 2026. Certified Speculative Execution for Untrusted AI Agents. arXiv:2606.31023. <https://arxiv.org/abs/2606.31023>
- [8] Kaixiang Wang, Yidan Lin, Jiong Lou, and Jie Li. 2026. BRA-Audit: Budgeted Runtime Auditing for LLM Multi-Agent Systems via Cumulative-Exposure Audit-Point Placement. arXiv:2608.14668. <https://arxiv.org/abs/2608.14668>
- [9] RM369. 2026. From Prompt-Level Goals to Prefix-Conditioned Execution: A Position Paper on Incremental Task Execution and Cumulative Audit. Preprint, Life of Research. <https://lifeofresearch.com/preprints/137>