

INDEPENDENT RESEARCH REPORT

# Kiodai: Structured Prospective Memory for Language-Model AI Agents and Its Evaluation Methodology

**Bagdat Beimzhan**

Independent Researcher

Astana, Kazakhstan · 2026

## Contents

|                                                                   |    |
|-------------------------------------------------------------------|----|
| Abstract.....                                                     | 3  |
| 1. Introduction.....                                              | 3  |
| 2. Theoretical Background and Related Work.....                   | 3  |
| 2.1. From Remembering to Executing an Intention.....              | 3  |
| 2.2. Related Memory Evaluations and Structured Stores.....        | 3  |
| 3. Systems and Implementation.....                                | 4  |
| 3.1. Historical and Final Conditions.....                         | 4  |
| 3.2. From Instruction to Execution Receipt.....                   | 4  |
| 4. Experimental Methodology.....                                  | 5  |
| 4.1. Scenarios, Freeze Procedure, and Run Conditions.....         | 5  |
| 4.2. Evaluator, Failures, and Unit of Analysis.....               | 6  |
| 5. Results.....                                                   | 6  |
| 5.1. Pilots and Functionality Check.....                          | 6  |
| 5.2. Main Comparison.....                                         | 7  |
| 5.3. Resource Use and Obligation Accounting.....                  | 7  |
| 6. Failure Analysis and Discussion.....                           | 9  |
| 6.1. Observed Failure Stages.....                                 | 9  |
| 6.2. Execution Chains and a Possible Explanation.....             | 9  |
| 7. Limitations, Practical Relevance, and Author Contribution..... | 10 |
| 8. Conclusion.....                                                | 10 |
| 9. References.....                                                | 10 |

## Abstract

An agent may retain a deferred instruction yet fail to act when its condition becomes true. This report evaluates prompting and structured intention tracking in a text-based prospective-memory simulator. Two initial pilots compared a baseline agent (A0) with the same agent plus a fixed instruction (A1); neither showed an improvement. A separate architectural comparison then tested A0, a structured-store variant with model-directed state queries (B\_ledger), and a variant with controller-directed queries (A2). DeepSeek V3.1 was evaluated once under each method on 12 development trajectories from four scenario families. Mean trajectory Set-F1 was 0.8643 for A0, 0.8111 for B\_ledger, and 0.4806 for A2. The paired mean A2 – B\_ledger difference was  $-0.330556$ . A2 did not improve action selection in this test. Saved traces show errors in intention extraction, revision, and action binding, including failure after a relevant observation was retrieved. All scenarios were exposed during development, and computation differed across methods. These results describe the tested implementations, not the general effectiveness of structured memory.

## 1. Introduction

Consider an instruction to archive a report only after a cooling board becomes stable. The agent receives other messages in the meantime. The condition may later be revised or cancelled, and archiving may depend on another task. Successful execution requires the agent to maintain the current obligation, inspect permitted evidence when needed, and select the correct action at the right checkpoint. Repeating the instruction in a later answer would not satisfy it.

This distinction is central to prospective memory. Retrospective memory concerns information about an earlier event; prospective memory concerns carrying out an earlier intention when its trigger occurs. PM-Bench tests this behavior in a text-based environment where agents continue an activity, encounter delayed obligations, and may query environmental state [1]. The present study uses its simulator and evaluation conventions, with separately constructed development scenarios.

Kiodai was developed in two stages. The first asked whether a fixed prospective-memory instruction (P1) helped a baseline agent. A0 and A1 differed by that instruction in paired pilots. After the pilots showed no observed gain, the project adopted a distinct architectural protocol: B\_ledger and A2 both maintained structured intention records, but differed in who selected state queries. B\_ledger left the choice to the model; A2 used a bounded programmatic controller [6]. These are separate experimental questions and are not pooled as one trial.

The primary architectural question was whether A2 achieved a positive mean trajectory Set-F1 difference relative to B\_ledger. A0 provided a secondary, simpler reference. The evaluation also recorded model usage, execution outcomes, and intermediate failures. The purpose is to explain what the implemented methods did under the recorded conditions, including failures that a single end-of-trajectory score can obscure.

## 2. Theoretical Background and Related Work

### 2.1. From Remembering to Executing an Intention

Rendell and Craik’s Virtual Week paradigm embeds future obligations in simulated everyday activity [2]. PM-Bench adapts this structure for language-model agents, with a scene, an action menu, conversation history, and channels for observing environmental state [1]. The analogy concerns the task structure; the agent experiments do not measure human memory or establish a psychological mechanism.

PM-Bench already evaluates agents with contextual todo-ledgers that describe intentions, conditions, and statuses [1]. The system and evaluator used here build on that environment. Because the final Kiodai scenarios were constructed separately and consulted during development, these scores must not be presented as new performance on PM-Bench’s released final week.

### 2.2. Related Memory Evaluations and Structured Stores

TriggerBench distinguishes prospective intervention from answering an explicit memory question. Its positive and negative cases, along with competing tasks, test whether an agent acts when a constraint applies and refrains

when it does not [3]. This motivates counting false actions as well as omissions. Its measures and scenarios are not directly interchangeable with the Set-F1 results reported here.

MemoryAgentBench tests retrieval, learning during interaction, long-range understanding, and selective forgetting [4]. Those capabilities are broader than the deferred-action process examined in Kiodai. The present experiment focuses on whether a maintained intention leads to an appropriate action at a scheduled checkpoint.

The Prospective Intention Store (PIS) separates intention formation and revision from programmatic lifecycle operations, using typed records and bounded model decisions [5]. Kiodai’s architectural stage reconstructs selected design ideas from PIS. It does not establish a faithful replication of that system or a controlled comparison with PIS results. The contribution here is the particular implementation, the frozen A0/B\_ledger/A2 comparison, and the inspection of recorded failures. Intention stores, versioning, and scheduled monitoring are not claimed as new concepts.

### 3. Systems and Implementation

#### 3.1. Historical and Final Conditions

A0 uses the baseline instruction, the history available to that agent, and permitted tools. A1 adds the fixed P1 instruction without introducing external intention storage. A1 was evaluated only in the pilots. B\_ledger and A2 belong to the later architectural protocol and share an Agent, Store, Gateway, extraction schema, and model prompts. Both support versioned records, dependencies, cancellation, binding validation, and receipt-based status updates [6].

Table 1. Experimental conditions. A1 was not included in the final architectural comparison.

| Feature                                         | A0               | A1 / P1                | B_ledger          | A2                |
|-------------------------------------------------|------------------|------------------------|-------------------|-------------------|
| Evaluation                                      | Pilots and final | Pilots only            | Final             | Final             |
| History and permitted tools                     | Yes              | Yes                    | Yes               | Yes               |
| P1 instruction                                  | No               | Yes                    | Separate protocol | Separate protocol |
| External intention store                        | No               | No                     | Same design       | Same design       |
| Stored revisions / dependencies / cancellations | No store layer   | No store layer         | Yes               | Yes               |
| Receipt-based store updates                     | No store         | No store               | Yes               | Yes               |
| Query selection                                 | Model            | Not evaluated in final | Model             | Controller        |

An agent without an external store is not deprived of its earlier instructions. Every final condition receives its own available history, current action menu, and prior action outcomes. Histories diverge after the methods make different observations or selections. A0 can make another model call after querying a channel; it is not restricted to one call per checkpoint.

#### 3.2. From Instruction to Execution Receipt

An early analytical representation used  $I = (c, a, s, v)$ , denoting condition, action, state, and version. The implemented records are more detailed: they contain an identifier, condition type and text, any available day and time, observation channel, dependencies, evidence quotations, status, version, and execution-attempt information. SQLite stores the records and events. A schema-valid record can still express the wrong instruction [6].

At a checkpoint, a model proposes record creations or revisions. Code checks their structure, references to visible observations, and allowed operations. An unresolved intention may be quarantined and excluded from execution. For A2, the monitoring controller considers channels connected to eligible unfinished intentions and prefers the channel least recently checked. Both structured conditions allow at most one environmental query per checkpoint. An additional clock query is unnecessary if the scene already exposes the relevant time.

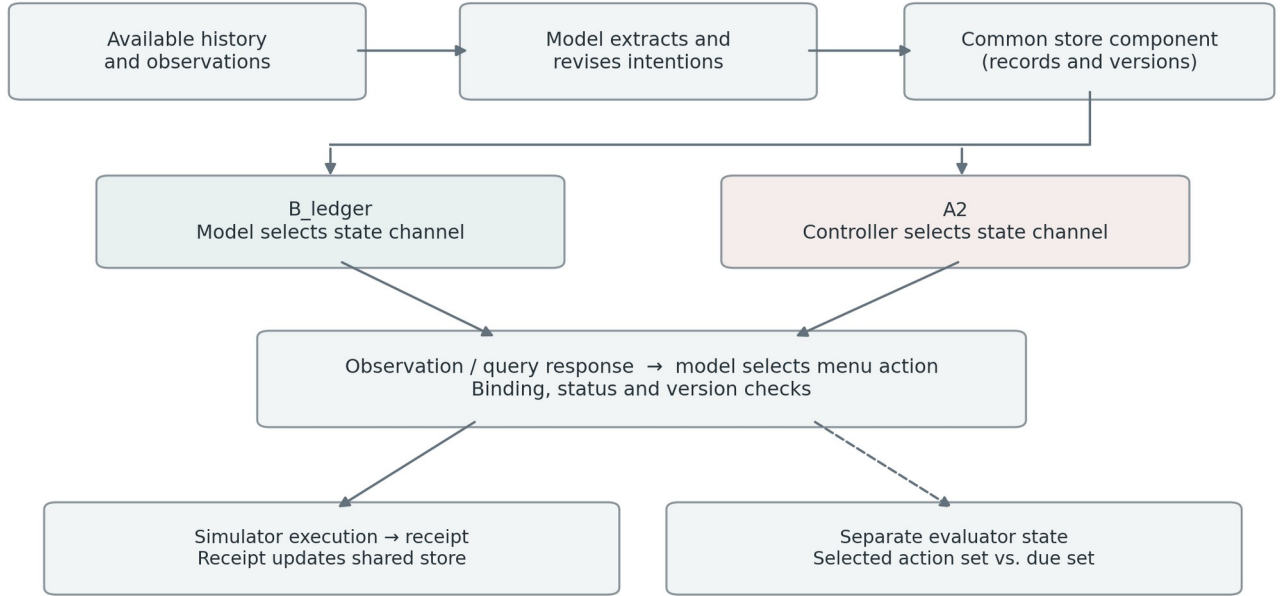


Figure 1. Intention processing in B\_ledger and A2. Extraction, storage, action selection, and receipt handling are shared; state-query selection differs. Evaluation uses separate state.

After observing the scene or a query response, the model selects an action-menu identifier and binds it to an intention version. Programmatic checks enforce status and dependency restrictions; they do not prove that the selected action matches the intended day, event, or semantic condition. A correct stored field such as “DayTwo, 11:00” is not by itself a guard against executing a distractor action on DayOne.

The simulator returns an execution receipt after a selection. Successful receipts mark the associated intention complete, while unsuccessful or uncertain outcomes are recorded separately. A wrong binding can attach a receipt to the wrong intention. Rejecting old version numbers also cannot prevent an obsolete deadline from being written back as a new version. Real-world exactly-once execution was not tested.

## 4. Experimental Methodology

### 4.1. Scenarios, Freeze Procedure, and Run Conditions

The frozen v2.1-comparison-v1 protocol contains 12 synthetic trajectories: three variants in each of four scenario families. The hidden family requires an environmental query; visible\_events uses visible event cues and revisions; cross\_day moves an obligation to another day; and revision includes rescheduling and cancellations. Each trajectory contains eight checkpoints. The cross\_day checkpoints span two simulated days. All scenarios were exposed during development and mock checks; v2\_hidden\_91320 also overlapped an earlier smoke test [6].

Each method ran once on each trajectory, producing 36 method-trajectory runs and 288 checkpoints. There were no repetitions, exclusions, or selective reruns. Family and variant order followed seed 20260910. Method order rotated cyclically across positions. Checkpoints were scheduled by the simulator. Heartbeat was disabled for every condition: A2 had no autonomous opportunities between checkpoints.

The candidate code was frozen at commit 959db38dac8b68f63ecf79420dcd53bea2278cf2; the 11 September 2026 experiment used cd0ce89e10d036918d1af06e5f5f2140930a0981. Experiment-support code was at 832d6ff. The source manuscript records a manifest of 59 source, configuration, and scenario hashes and an archive of 445 original run files. It reports rechecking their hashes and official scores during preparation [6, 7].

Table 2. Recorded configuration for the final comparison.

| Parameter          | Recorded setting                           |
|--------------------|--------------------------------------------|
| Model / identifier | DeepSeek V3.1; deepseek/deepseek-chat-v3.1 |
| Route              | OpenRouter / Novita; no route fallback     |

| Parameter                     | Recorded setting                                                                          |
|-------------------------------|-------------------------------------------------------------------------------------------|
| Generation                    | temperature = 0; top_p = 1; seed = 20260904                                               |
| Other request settings        | fp8; reasoning.enabled = false; reasoning.exclude = true; require_parameters = true       |
| Output limits                 | A0: 256 tokens; extraction: 3072; B_ledger/A2 selection: 1536                             |
| Available context and tools   | Full own history; at most one state query per checkpoint; heartbeat = false               |
| Validation / transport limits | One validation retry; zero transport retries; 120-second timeout; 48,000-byte request cap |

Saved requests identify the intended model, route, parameters, and limits, and their responses name DeepSeek V3.1 through Novita. They do not independently establish how the provider implemented each parameter. Compute was not matched: B\_ledger and A2 use separate extraction calls and larger output limits, and retries add to recorded resource use.

## 4.2. Evaluator, Failures, and Unit of Analysis

The agent sees its own interaction history, current scene, observations, permitted action menu, available channels, and prior action outcomes. Ground-truth due sets and canonical expected identifiers remain on the evaluator side. The runner does not use the original PM-Bench fallback that substitutes due actions after an unparseable reply. If an extraction or selection still fails validation after the allowed retry, no deferred target action is selected. Choosing the generic scene-continuation action “A” does not count as fulfilling an obligation [6].

The source review inspected the data boundary and saved requests and found support for isolation in the recorded runs. It does not establish that every conceivable leakage channel is impossible. Logged artifacts include raw requests and responses, validation retries, actions before and after identifier mapping, scores, token counts, durations, and provider-reported costs [7].

Due actions are calculated from evaluator state, using currently applicable obligations, time, events, cancellations, and recorded completions. This set can differ from what the agent believes is due. The evaluator can know a hidden cue even when the agent has not queried it. A failed prerequisite can prevent a dependent task from becoming due, so the number of obligations that ever become due may differ across methods.

For each trajectory, TP counts correctly selected due actions, FP counts extra actions, and FN counts omitted due actions. Set-F1 measures agreement between action sets, not the fraction of correct questions answered [1, 7]:

$$\begin{aligned}
 \textit{Precision} &= TP / (TP + FP) \\
 \textit{Recall} &= TP / (TP + FN) \\
 \textit{Set-F1} &= 2TP / (2TP + FP + FN)
 \end{aligned}$$

The primary summary is the arithmetic mean of the 12 trajectory-level Set-F1 values, giving each trajectory equal weight. The primary contrast is the mean paired difference A2 – B\_ledger, calculated from the 12 matched trajectories; A2 – A0 is secondary. Micro Set-F1 first pools TP, FP, and FN across trajectories, so trajectories with more countable actions carry more weight in that calculation.

The auxiliary calculation returns an undefined value if its denominator is zero; all 36 Set-F1 denominators here were positive. The source analysis reports no standard deviations, confidence intervals, or significance tests. Eight checkpoints within one trajectory and multiple model calls within a run are not independent replications. The 12 trajectories also share four templates.

## 5. Results

### 5.1. Pilots and Functionality Check

In the first pilot, A0 and A1 both reached Set-F1 = 1.00 (TP = 5, FP = 0, FN = 0). In the longer three-day follow-up, both reached 22/23 ≈ 0.9565 (TP = 11, FP = 0, FN = 1). A1 used more input tokens without increasing either score. The first pilot was at the metric ceiling, so these observations cannot establish that instruction changes are ineffective generally [8].

An earlier v2 development check failed with an empty intention store. After repairs, an eight-checkpoint v2.1 smoke test completed three intentions (TP = 3, FP = 0, FN = 0; Set-F1 = 1.00), using seven monitoring queries and receiving three successful execution receipts. It was a functionality check on a known scenario, not a controlled comparison. Field errors still occurred and were corrected within that run [9].

## 5.2. Main Comparison

A0 had the highest primary score. A2 performed below B\_ledger despite using the same structured store and differing chiefly in state-query policy. Table 3 reports both trajectory means and micro-aggregated scores; the two aggregates are not expected to be identical [7].

Table 3. Final comparison (12 trajectories per method). TP, FP, and FN are pooled counts; the last column averages trajectory-level Set-F1.

| Method   | TP | FP | FN | Micro Set-F1 | Mean trajectory Set-F1 |
|----------|----|----|----|--------------|------------------------|
| A0       | 22 | 6  | 2  | 0.8462       | 0.8643                 |
| B_ledger | 18 | 4  | 3  | 0.8372       | 0.8111                 |
| A2       | 12 | 5  | 7  | 0.6667       | 0.4806                 |

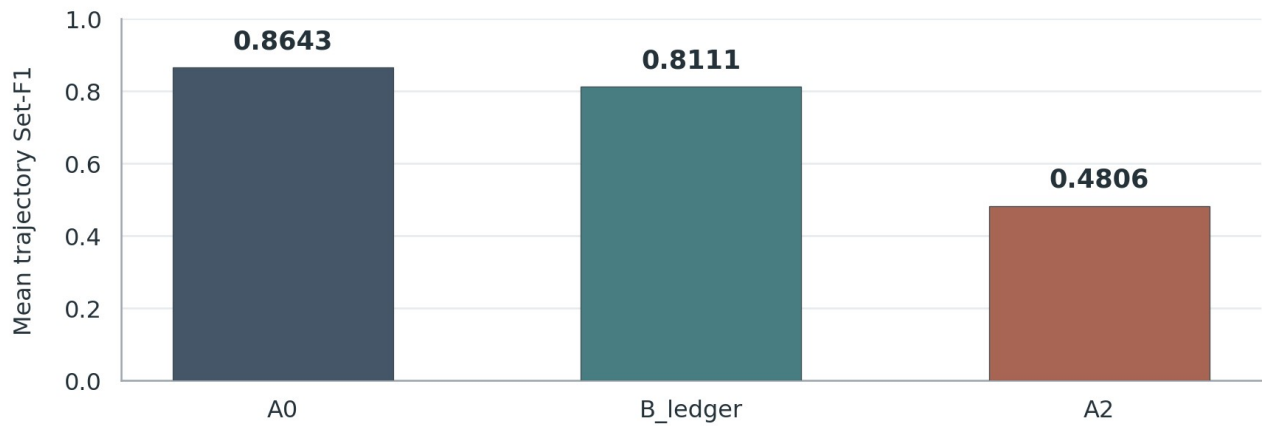


Figure 2. Mean trajectory Set-F1 in the frozen comparison. All bars start at zero; dispersion was not estimated, so no uncertainty intervals are shown.

The mean paired difference was  $-0.330556$  Set-F1 units for A2 – B\_ledger and  $-0.383730$  for A2 – A0. These values were computed in the source analysis from trajectory-level scores rather than differences between four-decimal table entries. Neither comparison supports the predicted A2 improvement in this sample. No significance test was reported.

A2's mean varied by scenario family: 0.0000 in cross\_day and 0.8667 in visible\_events. B\_ledger reached 0.8000 in visible\_events, where A2 made no tool queries; the difference in that family cannot demonstrate a monitoring benefit. On v2\_hidden\_91320, which overlapped the smoke test, A2 scored zero despite having a non-empty store and making six queries. Historical pilots and smoke tests were not counted as further replications [7, 9].

## 5.3. Resource Use and Obligation Accounting

Table 4. Model calls, token usage, and API-reported cost for the recorded final comparison.

| Method   | Calls | Input tokens | Output tokens | Cost (USD) |
|----------|-------|--------------|---------------|------------|
| A0       | 117   | 177,855      | 3,175         | 0.04047    |
| B_ledger | 251   | 910,740      | 44,534        | 0.21212    |
| A2       | 234   | 843,540      | 42,889        | 0.19871    |
| Total    | 602   | 1,932,135    | 90,598        | 0.45130    |



Figure 3. API-reported cost of the 36-run comparison, by method. Values are for this run, not current provider prices or total project spending.

Using unrounded records, A2 cost USD 0.19870816 and A0 cost USD 0.04047361, a ratio of approximately 4.91. The source reports USD 0.45130261 across all final runs. Provider-reported charges include extraction, selection, and retries, but were not reconciled to a final invoice. Token usage and cost should not be read as measurements of latency [7].

Table 5. Environmental queries, action selections, and validation outcomes in the final comparison.

| Method   | Hidden-board queries | Query-supported successes | Executions | Successful receipts | Selection-validation failures |
|----------|----------------------|---------------------------|------------|---------------------|-------------------------------|
| A0       | 15                   | 6                         | 28         | 22                  | 2                             |
| B_ledger | 8                    | 4                         | 22         | 18                  | 24                            |
| A2       | 18                   | 3                         | 17         | 12                  | 15                            |

A query-supported success denotes an action selected with a relevant state-channel response available at that checkpoint. It documents compatible evidence, not proof that the query caused the model’s choice; one response can support several actions. Selection-validation failures count attempts, including retries. Dividing them by all model calls would give a misleading rate because many calls were extraction requests and the selection-attempt denominator was not tabulated.

Table 6. Original obligations traced through evaluator state (27 initially instructed per method).

| Method   | Instructed | Ever due | Dependency blocked | Completed | Unfinished* | Cancelled |
|----------|------------|----------|--------------------|-----------|-------------|-----------|
| A0       | 27         | 24       | 0                  | 22        | 2           | 3         |
| B_ledger | 27         | 21       | 3                  | 18        | 6           | 3         |
| A2       | 27         | 19       | 5                  | 12        | 12          | 3         |

\* “Unfinished” excludes cancelled obligations. Completed + unfinished + cancelled = 27 in each method.

Three cancellations in the revision family account for the remainder in every condition. A0’s zero dependency-blocked count was reconstructed in the source analysis rather than assumed. A task is classified as blocked when its trigger occurs while a prerequisite remains unfinished [7].

Five dependent A2 obligations never became due, two in hidden and three in cross\_day. A2 therefore had 12 unfinished non-cancelled obligations but only seven official false negatives. An earlier failure can suppress later due obligations; the smaller due-set count is not evidence of better performance. Table 6 records this difference without changing the benchmark metric.

## 6. Failure Analysis and Discussion

### 6.1. Observed Failure Stages

The post-run semantic review covered the 24 structured-method trajectories; 12 A0 trajectories were reviewed separately. AI tools assisted the annotations, and no independent human annotator or inter-rater agreement measure was used. Table 7 lists distinct symptoms and their earliest documented stages. It is a set of examples, not a frequency distribution or a partition of failures [7, 10].

Table 7. Documented failure examples. Scenario labels have the prefix v2\_; trace references follow the source review.

| Symptom                            | First evidenced stage                                 | Consequence                                    | Trace and interpretation limit                                                             |
|------------------------------------|-------------------------------------------------------|------------------------------------------------|--------------------------------------------------------------------------------------------|
| Missing record at deadline         | Extraction batch rejected over a dependency reference | Registration missed                            | A2 hidden_91320, checkpoints 1–3. Drafts named tasks; lack of understanding is not proven. |
| Dependency omitted                 | Accepted extraction lacked a prerequisite             | Eligibility changed; later task may be blocked | A2 hidden_91320, checkpoints 3–4. Underlying cause not isolated.                           |
| Old deadline returned              | New revision stored an obsolete time                  | Premature action; failed receipt               | A2 revision_91300, checkpoints 4–7. Revision and selection failures coexist.               |
| Future day recorded, action now    | Selection bound a distractor menu item                | False action; required registration unfinished | A2 cross_day_91331, checkpoint 3. Stored day was correct.                                  |
| Positive board evidence, no action | Selection validation failed after a query             | Empty target-action set                        | A2 hidden_91320, checkpoint 7. Lack of queries does not explain this step.                 |
| Superseded event cue returned      | Accepted condition revision                           | Premature sealing                              | A2 visible_events_91310 and B_ledger visible_events family. Based on AI annotation.        |

The source review checked the first five examples against stored responses, snapshots, and receipts. The last example depends on an AI annotation with pointers to raw responses. The earlier v2 extraction-contract defect had been repaired before the final comparison, so its existence alone cannot explain the subsequent score gap [7, 10].

### 6.2. Execution Chains and a Possible Explanation

In A2’s v2\_hidden\_91320 run, extraction batches were rejected at checkpoints 1 and 2 and the store remained empty. Records appeared at checkpoint 3, after the registration deadline. The dependency needed for sealing was initially missing and later restored. At checkpoint 7, board\_cobalt91320 returned positive state evidence, but action selection and its permitted retry failed validation. No target action or execution receipt followed. Relevant information had been obtained; the failure occurred downstream [7, 10].

In v2\_revision\_91300, A2 wrote the obsolete 12:00 deadline into a new version after the obligation had moved to 13:00. It then attempted the action too early at checkpoint 6 and received simulator\_not\_completed. At checkpoint 7 it restored 13:00 and received simulator\_completed. All 16 responses in this trajectory passed structural validation. Valid syntax and valid references did not guarantee correct revision semantics [7, 10].

In v2\_cross\_day\_91331, the record correctly specified DayTwo. At checkpoint 3 on DayOne, the agent selected a distractor action for a spare sample and bound it to the registration record. The receipt was unsuccessful, and registration remained incomplete. The stored day was not the earliest evidenced error; action selection and binding were [7, 10].

The pipeline could also complete the tasks. In v2\_hidden\_91322, A2 registered after correcting a dependency and retrying selection. At checkpoint 7, it used positive board evidence to select sealing and archiving, which both received successful receipts. That trajectory ended with Set-F1 = 1.00 despite intermediate mistakes. A0 completed the same three obligations with fewer queries and model calls [7, 10].

These traces are compatible with semantic errors propagating through intermediate state: a wrong dependency or deadline can alter eligibility, and a wrong binding can prevent completion despite a correct record. The experiment did not isolate extraction, prompt design, controller policy, validation, or model behavior as the cause of A2’s overall deficit. B\_ledger recorded more selection-validation failures than A2 (24 versus 15) while scoring higher. A simple count of formatting failures does not explain the performance ranking.

## 7. Limitations, Practical Relevance, and Author Contribution

The final evaluation used one DeepSeek V3.1 configuration, one run per method-trajectory pair, and 12 trajectories drawn from four shared templates. Scenarios were consulted during development, including one in smoke testing. This is not a held-out evaluation. Fixed seeds and zero temperature do not replace repetitions or ensure that a remote provider returns identical outputs. There is no reported uncertainty estimate from which to infer general performance differences.

The conditions also used unequal model computation. A0 has a different call structure and shorter output limit than the structured methods. B\_ledger and A2 share most components but may observe different information and acquire different histories after query selection. A2 versus A0 is a whole-system contrast; A2 versus B\_ledger is a comparison of two policies in a shared architecture. Neither estimates a universal causal effect of adding external memory.

AI-assisted semantic annotation lacked a separate human review panel. The source review examined selected raw traces, but alternative explanations remain possible. Provider responses reported token usage and charges; an invoice and provider-side implementation details were not independently verified. Frozen artifacts make the recorded evaluation auditable, but replaying requests may yield different responses.

Tasks and observations were in English in a simulated environment. The work did not test multilingual interaction, real background execution, external side effects, user benefit, or exactly-once delivery. PM-Bench removes completed actions from the menu, which also prevents an independent test of protection against repeating an action after completion. These limits matter before applying the findings to scheduling or organizational assistants.

The author framed the prospective-memory question and specified the comparison protocol as documented in the study materials. AI tools, including Codex, were used for programming, log analysis, and manuscript preparation. Independent authorship of every code line and independent human annotation are not claimed. PM-Bench and its evaluator are credited to their original authors, and the intention-store ideas are attributed to PIS and related work. These attributions do not change the reported empirical results.

## 8. Conclusion

Neither initial A0/A1 pilot showed an improvement from the fixed instruction. In the later frozen comparison, A2 reached a mean trajectory Set-F1 of 0.4806, below B\_ledger (0.8111) and A0 (0.8643), while A0 incurred the lowest recorded API cost. The A2 improvement hypothesis was not supported on these 12 development trajectories. Trace analysis located failures in extraction, revision, and action binding; the experiments did not identify a single causal bottleneck or disprove structured intention storage in general.

A diagnostic follow-up could replace model-produced extraction and revision with reference annotations constrained to information actually visible at each checkpoint, while leaving downstream steps unchanged. It would test whether incorrect record formation explains part of the deficit. Future facts, hidden answers, and unobserved completions must be excluded. This intervention has not been run and would not, by itself, demonstrate deployable performance or establish an automatic upper bound.

## 9. References

### External literature

- [1] Liu, G., and Gabriel, S. “PM-Bench: Evaluating Prospective Memory in LLM Agents.” arXiv:2607.12385v1 (14 July 2026); COLM 2026. <https://arxiv.org/abs/2607.12385v1>.
- [2] Rendell, P. G., and Craik, F. I. M. “Virtual Week and Actual Week: Age-related Differences in Prospective Memory.” *Applied Cognitive Psychology* 14(7) (2000): S43–S62. <https://doi.org/10.1002/acp.770>.
- [3] Zhang, T., Wang, X., Zhang, Q., Chen, Q., Li, K., Chen, Y., Wang, D., Meng, H., and Lu, Y. “TriggerBench: Investigating Prospective Memory for Large Language Models.” arXiv:2606.23459v1 (22 June 2026). <https://arxiv.org/abs/2606.23459v1>.
- [4] Hu, Y., Wang, Y., and McAuley, J. “Evaluating Memory in LLM Agents via Incremental Multi-Turn Interactions.” ICLR 2026; arXiv:2507.05257v4 (revised 28 June 2026). <https://arxiv.org/abs/2507.05257v4>.

- [5] Zhao, J., and Wu, C. "Making Prospective Memory SLM-Shaped: Typed Intention Stores for Small-Model Agents." arXiv:2609.01272v1 (1 September 2026). <https://arxiv.org/abs/2609.01272v1>.

### **Project records and artifacts**

- [6] Kiodai. Frozen protocol and implementation, v2.1-comparison-v1 (10–11 September 2026). `research/v2_1/comparison_v1.json`; `configs/v2_1_comparison_v1.json`; `kiodai_v2/agent.py` and `store.py`; `prompts/v2_1/`. Commit identifiers appear in Section 4.1.
- [7] Kiodai. Primary records and final scores, v2.1-comparison-v1 (11 September 2026). `results/v2_1/comparison-v1/`: `scenario.json`, `calls.jsonl`, `actions.jsonl`, `steps.jsonl`, and `score.json` across 36 run directories; `comparison_report_reviewed.json`; `research/v2_1/live_comparison_v1_inventory.json`.
- [8] Kiodai. A0/A1 pilot records (8–9 September 2026). Historical manuscript, `RESULTS.json`, `FOLLOWUP_RESULTS.md`, `FOLLOWUP_RESULTS.json`, `live-pair-42433e00474b`, and `live-pair-bdf241965f35`.
- [9] Kiodai. v2 development and separate v2.1 smoke test (9 September 2026). `docs/v2/DEEPSEEK_SMOKE_RESULTS.md`; `docs/v2_1/DEEPSEEK_SMOKE_RESULTS.md`; `results/v2_1/deepseek-smoke-v1/analysis.json`.
- [10] Kiodai. Post-run failure review (11 September 2026). `results/v2_1/comparison-v1/semantic_review.json` and `baseline_behavior_review.json`; `docs/v2_1/COMPARISON_RESULTS.md`. AI-assisted annotations and pointers to raw traces.

Repository: <https://github.com/Bagnesium/Kiodai>