STREAMING INTENT TRANSACTIONS FOR SAFE ASYNCHRONOUS TOOL USE IN FULL-DUPLEX VOICE AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Full-duplex voice agents can dispatch tools before a user finishes speaking, reducing latency but creating a temporal authorization problem: a callback may return after the user has revised or revoked the intent that authorized it. Existing evaluations primarily measure routing and tool correctness rather than whether a delayed result remains authorized to change external state. We formalize this failure mode as *streaming intent transactions* and propose DUPLEXTXN. A causal policy over a frozen Omni backbone compares dispatch- and commit-time representations, while a deterministic gate verifies version, binding, dependency, confirmation, and idempotency constraints. We also introduce DUPLEXTXN-Bench, a paired-timing evaluation that holds speech, tool output, and initial state fixed while moving callback arrival across a spoken revision boundary. Across 516 synthetic out-of-distribution families and four independently trained seeds, DUPLEXTXN reduces unsafe-effect episodes from 17.05% to 0.024%, eliminates observed stale-result effects, and raises safe task completion from 8.58% to 82.63% relative to our execution of released DuplexOmni. On a post-freeze natural-speech transfer panel, the same frozen policies reach 100% safe completion under controlled callback replay. These results support temporal authorization as a distinct requirement for safe full-duplex tool use.

1 INTRODUCTION

Full-duplex voice agents interleave listening, speaking, reasoning, and tool use on one wall-clock timeline. DuplexSLA co-decodes speech and actions, while DuplexOmni runs a low-latency interaction path alongside asynchronous thinking and tools (Zhang et al., 2026; Huang et al., 2026). Such concurrency improves responsiveness, but invalidates an implicit assumption of turn-based tool use: the request remains stable until an effect is executed. In streaming interaction, however, users may revise, replace, or revoke an intent while an earlier tool call remains in flight.

To illustrate, consider two concurrent requests: an agent prepares a 10:00 flight and a hotel booking, then the user says, "The hotel—no, leave it; don't book the flight."

A hotel callback that arrives after "The hotel—no" but before the repair is resolved should be held, not committed or cancelled. After the complete correction, the hotel remains authorized while the flight does not. No second flight call is required, and "most recent call wins" is invalid because the two calls belong to different transactions. Cancelling after execution is already too late. This distinction between a valid result and a currently authorized effect is the central focus of this paper. Figure 1 traces the episode on the interaction timeline.

Prior work studies delayed feedback, symbolic futures, speculative calls, and transactional runtimes (Shi et al., 2026; Feng et al., 2026; Hooper et al., 2026; Chen et al., 2026). However, full-duplex speech adds an additional source of non-monotonicity: the authorizing intent evolves before a turn-final transcript exists. A safe agent must therefore satisfy two distinct obligations: infer whether subsequent speech changes the mandate, and enforce the authorization predicates that structured runtime state can verify exactly. Assigning both obligations to a single language-model decision entangles semantic errors with protocol violations.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

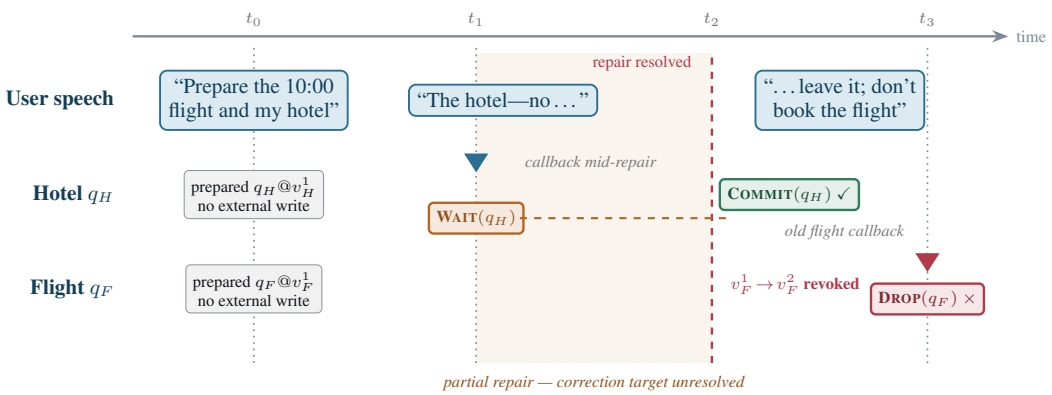

Figure 1: Why structured freshness alone is insufficient. Flight and hotel preparations are concurrent, not successive versions of one global request. When the hotel callback arrives, the user's partial repair is ambiguous, so DUPLEXTXN holds it with WAIT. The completed utterance preserves the hotel mandate but revokes the flight mandate; the deterministic gate then commits the hotel candidate and drops the stale flight candidate. If a structured event already stated that one call superseded another, counters would suffice. The learned component is needed here to infer from causal speech which transaction changed and when the repair became resolved; the runtime only enforces that inferred version and binding. No protected effect is written before COMMIT.

We introduce *streaming intent transactions*. Each protected effect is bound to the intent version that authorized its dispatch. A learned causal policy detects semantic change and compares dispatch-time with commit-time state. A deterministic boundary then enforces explicit transaction invariants. This separation is deliberate: the model handles ambiguous speech; the runtime handles version, binding, dependency, confirmation, and idempotency constraints.

We make three contributions:

1. We define commit authorization under non-monotonic spoken intent and separate the ledger-relative runtime invariant from the learned problem of semantic invalidation.
2. We propose DUPLEXTXN, a lightweight extension to a frozen full-duplex Omni agent that combines causal intent tracking, paired dispatch–commit comparison, and an atomic commit gate.
3. We introduce an executable paired-timing benchmark that isolates callback timing and jointly measures wrong commits and safe task completion, preventing refusal-only systems from appearing safe. We further provide a post-freeze external stress test built from held-out human-to-human SpokenWOZ audio.

Controlled synthetic experiments evaluate the proposed mechanism, while a separate outcome-blind SpokenWOZ panel evaluates frozen policies on natural human speech without training or reselection. We interpret these results as controlled evidence of transaction safety and acoustic transfer, not as unconstrained full-duplex generalization.

2 RELATED WORK

Full-duplex speech and Omni agents. SyncLLM and Moshi establish synchronous and parallel-stream foundations for full-duplex spoken dialogue (Veluri et al., 2024; Défossez et al., 2024). MiniCPM-o 4.5 places audio, vision, language, and speech generation on a streaming timeline, and Chronological Thinking studies strictly causal reasoning during listening (Cui et al., 2026; Wu et al., 2026). Freeze-Omni connects speech input and output modalities to a frozen language model, OmniFlatten progressively post-trains flattened speech and text streams, and ELLSA routes speech and actions through specialized experts (Wang et al., 2025; Zhang et al., 2025; Wang et al., 2026b). DuplexSLA adds a synchronized action channel, whereas DuplexOmni decouples interaction from asynchronous thinking and tools (Zhang et al., 2026; Huang et al., 2026). A recent frontend–backend

design trains a duplex speech frontend to emit a delegation token, sends streaming transcripts to a text tool agent, and injects the returned result back into speech generation (Hu et al., 2026). Its frontend remains silent during backend execution; the work evaluates delegation and result reinjection rather than commit-time authorization. We analyze commit-time authorization when later speech may invalidate the authority carried by a delayed result. Realtime-Venus is a closely related concurrent system: its dual-loop harness freezes request-boundary evidence, supports cancellation and replanning supervision, and excludes results that miss a freshness deadline (Zhao et al., 2026). Its reported delegation benchmark measures routing decisions, and the authors explicitly leave successful execution and conversational reintegration for further evaluation. In contrast, our target is semantic authorization at the durability boundary: whether a delayed result may still create an external side effect after the spoken intent has changed. We therefore learn callback-time intent comparison, combine it with deterministic transaction predicates, and evaluate durable stale, duplicate, missed, and safe-completion outcomes under controlled callback schedules.

Parameter-efficient streaming adaptation and runtime shields. The architectural pattern of freezing a general backbone and learning a small causal state tracker is established outside tool use. Streaming Detection of Queried Event Start adapts frozen vision–language backbones with reduced-width temporal modules that retain state across an unbounded video stream and make prefix-only event decisions (Eyzaguirre et al., 2024). AdaTAD++ similarly places independently trainable spatial and temporal adapters around pretrained video backbones and uses a Transformer to capture long-range temporal dependencies (Agrawal et al., 2025). These methods motivate our representation strategy, but they predict perceptual events rather than whether an earlier authorization remains valid. A complementary precedent comes from robotics: Latent Safety Filters learn a safety representation in a world model's latent space and use it to shield an otherwise arbitrary task policy (Nakamura et al., 2025). DUPLEXTXN follows the same division of labor between learned semantic judgment and an execution-time boundary, while making a narrower guarantee: the deterministic gate enforces only predicates recorded in the transaction ledger; speech-level invalidation remains a learned and empirically evaluated obligation.

Full-duplex tool-use evaluation. τ-bench evaluates policy-constrained agents against final database state, and τ-Voice extends executable tasks to full-duplex audio (Yao et al., 2024; Ray et al., 2026). FDB-v3 combines human disfluencies and chained tool calls across four domains (Lin et al., 2026). HumDial-FDBench studies overlap, interruption, and conversational flow, while Omni-Interact measures correctness, timing, interruption, and continuity under native online audio-visual inference (Wang et al., 2026a; Lu et al., 2026). Full-duplex tool use and correction robustness are therefore not contributions of this work. DUPLEXTXN-Bench instead intervenes on the arrival time of an identical callback relative to a spoken authorization boundary and scores the resulting durable state.

Asynchronous tool use. AsyncTool evaluates delayed callbacks, task switching, and dependencies; AsyncFC represents unresolved calls as symbolic futures (Shi et al., 2026; Feng et al., 2026). Speculative Interaction Agents revise calls made from partial input and hold unsafe writes until the query and plan settle (Hooper et al., 2026). AgentChangeBench measures goal shifts, redundant calls, and recovery (Rana et al., 2025). These approaches operate on text or endpoint behavior. Our method consumes native streaming Omni states and rechecks the concrete intent version at durability. This scope does not solve issue-time privacy: once an external service sees a speculative request, a later commit gate cannot retract the disclosure (Mohammadi et al., 2026a).

Transactions and temporal authorization. Cordon stages task-scoped effects with lineage, Atomix combines transaction sealing with progress-gated commit, and ACRFence prevents duplicate or resurrected effects after restore (Chen et al., 2026; Mohammadi et al., 2026b; Zheng et al., 2026). Related systems revalidate temporary authority, provenance-bearing memory, predecessor state, and dependency structure at commit time (Santos-Grueiro, 2026; Li et al., 2026; He & Yu, 2026; Salas, 2026). SafeCommit instead uses calibrated possible-world certificates in a controlled mechanism study (Akewar & Ranjan, 2026). DUPLEXTXN adopts the general separation between planning and effect commitment, but adds the learned temporal link from continuous audio to a concrete transaction version.

3 PROBLEM FORMULATION

An interaction is a wall-clock event sequence $H = \langle e_1, \ldots, e_T \rangle$ containing audio chunks, assistant speech, intent hypotheses, tool dispatches, callbacks, confirmations, and effects. At time t, the active intent snapshot $I_t = (v_t, z_t, a_t)$ contains an authorization epoch v_t, semantic hypothesis z_t, and authorization state a_t. Successive hypotheses may preserve, extend, correct, replace, cancel, or confirm the active mandate. Parallel requests, backchannels, and unrelated speech remain protocol-level evaluation relations because the frozen training split contains no positive support for learning them.

Each tool dispatch creates an immutable ticket q that binds the authorizing epoch, target, payload, dependencies, confirmation requirement, and idempotency key. A *wrong commit* occurs when an effect is externalized although the current mandate no longer authorizes that concrete ticket. A *missed effect* occurs when a valid, ready, and authorized ticket is not committed. Safety alone is therefore insufficient: a system that blocks every write has no wrong commits but provides no task utility.

The durability boundary is essential. In our protocol, asynchronous dispatch may read, compute, reserve a tentative result, or prepare a mutation, but it may not externalize a protected effect; only COMMIT may do so. Consequently, DROP means "discard an uncommitted candidate," not "undo a completed write." If an integration executes a mutation before the gate, it lies outside the protected protocol. If the user revises the mandate after a valid commit, the runtime must open a new compensating transaction (for example, update the 10:00 meeting to 3:00, or cancel and recreate it). Such post-commit repair can itself fail or expose intermediate state and is not covered by the no-stale-commit guarantee evaluated in this paper.

There is also an information limit on commit-only classifiers. Let $o_c = (s_c, x_c)$ denote the policy observation when a callback is considered, excluding its dispatch-time semantic state. Consider two histories with the same o_c, candidate result, and executor facts, but where the candidate was dispatched under different spoken mandates: one remains authorized and the other has been replaced. The correct decisions differ, yet any deterministic classifier $f(o_c)$ must return the same decision for both histories and therefore err on at least one. Dispatch-conditioned state or equivalent lineage information is thus necessary to identify this pair; it is not merely an increase in model capacity. This argument does not guarantee that a learned paired policy is correct—it only makes the relevant distinction observable.

We separate two obligations. First, the learned policy must recognize semantic invalidation from causal speech. Second, the runtime must enforce every observable ledger predicate atomically at commit. Under complete event observability, unforgeable tickets, and atomic check-and-commit, the gate cannot commit a ticket that violates a recorded predicate. End-to-end stale-effect safety additionally requires the learned policy to revoke the old epoch before commit whenever speech changes the mandate. Appendix E formalizes this decomposition and the finite-state conformance check.

4 METHOD

Figure 2 summarizes DUPLEXTXN. We freeze the full-duplex backbone and train a lightweight policy over its causal streaming states. The design has three stages: track the evolving mandate, compare the states at dispatch and commit, and enforce structured authorization before an effect leaves the runtime.

4.1 TRACKING A NON-MONOTONIC MANDATE

For audio chunk t, the frozen backbone produces a native streaming state h_t. We concatenate six executor-observable features x_t indicating tool, result, dependency, speaking, irreversibility, and elapsed-time status. A two-layer, four-head causal Transformer of width 368 maps the prefix $(h_1, x_1), \ldots, (h_t, x_t)$ to a transaction state s_t. Causal masking prevents the policy from reading future audio during training. The output is one of 41 transaction states observed in the training protocol. Each state is a valid joint assignment of intent relation, semantic change, response schedule,

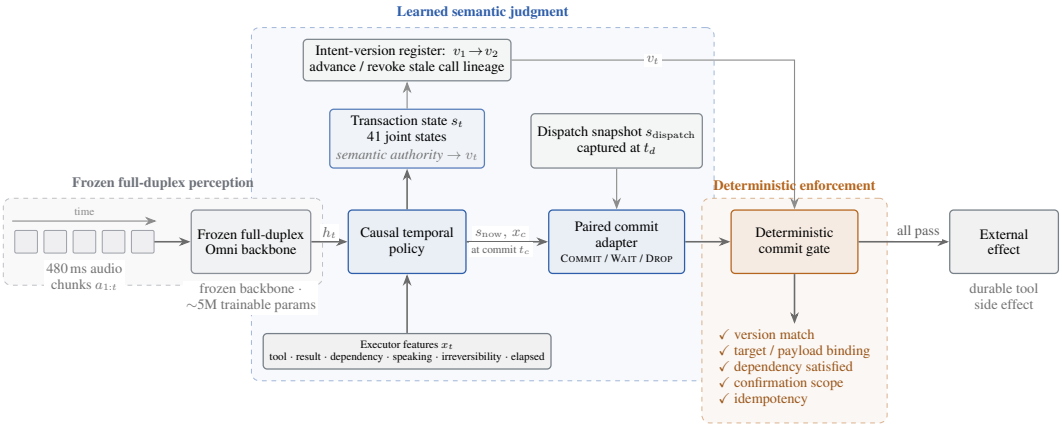

Figure 2: DUPLEXTXN architecture. A frozen full-duplex Omni backbone turns streaming 480 ms audio chunks into causal states h_t; a lightweight causal temporal policy (2 layers, 4 heads, width 368) tracks the non-monotonic mandate, emitting a 41-way transaction state s_t and a semantic-authority signal that advances or revokes the intent version. When a callback returns, the paired commit adapter compares the dispatch-time snapshot s_{dispatch} with the current state s_{now} and callback context x_c, proposing COMMIT/WAIT/DROP. A deterministic gate then checks version, binding, dependency, confirmation, and idempotency against the transaction ledger before any durable effect leaves the runtime. Only the policy and adapter are trained; the backbone and background tool agent stay frozen.

transaction action, and base commit decision. Joint decoding avoids contradictory combinations such as predicting cancellation and commit at the same instant.

A separate semantic-authority head predicts whether new speech preserves or invalidates the active mandate. An invalidating prediction advances the intent version and revokes older call lineage. Several calls may share one version, but each retains its own immutable target and payload. The model therefore learns the semantic question that the executor cannot derive from structured state: whether the user's mandate has changed.

The outputs are complementary rather than repeated labels. The joint state s_t controls the online dialogue–transaction protocol; the semantic-authority prediction decides whether the runtime advances or revokes an intent version; and the paired adapter below decides what to do with one concrete callback. Only the last component proposes COMMIT, WAIT, or DROP, and none can bypass the deterministic gate.

4.2 COMPARING DISPATCH AND COMMIT STATE

A callback is evaluated at an event boundary, not as a new audio chunk. For a ticket dispatched at time d and considered for commit at time c, the paired adapter receives

$$u_{d,c} = [s_d; s_c; s_c - s_d; s_c \odot s_d; x_c]. \tag{1}$$

One head predicts COMMIT versus BLOCK; conditional on blocking, a second predicts WAIT versus DROP. Both use categorical argmax. A current-state classifier cannot distinguish two identical callback states with different authorizing histories. The paired representation can: the call may have been valid at dispatch and stale at return.

4.3 ENFORCING THE COMMIT BOUNDARY

Every tool dispatch creates a ticket $q = (v_q, g_q, p_q, D_q, c_q, k_q)$ containing the authorizing intent version, target, payload hash, dependency set, confirmation requirement, and idempotency key. At commit time t, the executor accepts a candidate effect only when the learned policy requests commit and

$$G_t(q) = [v_q \in L_t] \wedge \text{bound}(q) \wedge \text{ready}(q) \wedge \text{confirmed}(q) \wedge [k_q \notin C_t], \tag{2}$$

where L_t is the set of live intent versions and C_t the committed-key set. The check, effect invocation, and local idempotency record form one atomic operation. The learned policy may request a commit, but it cannot bypass $G_t(q)$. Conversely, the gate cannot repair a missed semantic invalidation if the policy leaves an obsolete epoch live. This boundary yields a precise claim: violations of recorded predicates are prevented by construction, while semantic failures remain measurable model errors.

4.4 Training objective and data

The policy is trained on causal prefixes from the frozen backbone, not turn-final transcripts. Supervision covers joint transaction state, change boundaries, and semantic authority; a paired adapter then predicts commit eligibility and WAIT versus DROP. The 5,000-family corpus contains 10,600 utterance segments and 10,000 paired-timing scenarios, split by family and speaker into 3,976/508/516 train/calibration/synthetic-OOD families. The final adapter trains on 9,899 label-blind commit pairs, updates 463,824 parameters, and leaves the backbone and base temporal policy frozen. Four fresh seeds (911, 1013, 1217, 1429) are evaluated; selection uses training and calibration only, first constraining unsafe commits and then minimizing misses.

5 DuplexTxn-Bench

DuplexTxn-Bench estimates the causal effect of callback timing relative to a spoken revision boundary: the same callback should produce different durable outcomes before and after the user revises its authorizing intent. Each scenario contains time-aligned audio, intent events, calls, callbacks, confirmations, and a resettable executable environment. Outcomes are reconstructed from final environment state, not from assistant text or self-reported success.

5.1 Paired timing intervention

Every semantic scenario yields a matched pair. Both runs share the waveform, model context, tool arguments, tool result, initial database, and decoding seed. Only callback timing changes: one callback arrives before a spoken correction or cancellation, the other after it. This intervention isolates temporal authorization from speech content, acoustics, and tool quality. A run counts as complete only when its event trace agrees with the final environment snapshot; missing, failed, and timed-out runs remain in the denominator.

5.2 Data partitions

The corpus balances Chinese and English across five tool domains and five update relations. Paired schedules yield 10,000 episodes and 223,463 causal rows after prespecified removal of post-event no-ops. Speech comes from license-tracked, intelligibility-audited prompts; OOD families and lexical templates are disjoint from training and calibration. No human recording affects training, calibration, architecture, checkpoint, or threshold selection.

This design deliberately trades ecological validity for causal control. The paired waveform and executable clock isolate whether callback timing crosses a revision boundary, but they do not reproduce the full variability of human accents, prosody, overlap, microphone channels, or spontaneous reformulation. Public human-speech benchmarks such as FDB-v3 remain related-work comparators, not evidence used by this version of the paper (Lin et al., 2026).

Post-freeze SpokenWOZ panels test controlled natural-speech transfer: one fixes calls and replays callbacks around a revision boundary; the harder panel uses model-originated calls on 149 held-out families. Neither panel affects training, tuning, or selection; Appendix A gives the full protocol.

6 Experiments

6.1 Setup

We ask whether DuplexTxn improves safety without refusing valid work (RQ1), why dispatch state is needed (RQ2), whether its output can be simplified (RQ3), and how the frozen policy trans-

fers to natural speech and compares with turn-final controls (RQ4–RQ5). The principal comparison fixes the DuplexOmni checkpoint, tool agent and registry, audio, callback schedule, initial environment, and decoding; DUPLEXTXN adds only the temporal policy and deterministic gate. Every attempt remains in the denominator.

Baseline provenance. "Released DuplexOmni (our run)" means our execution of the public author code and checkpoint inside the shared benchmark harness; it is not an author-reported number. Named "inspired" controls in the appendix are our bounded mechanism-level implementations over a shared event stream, not official or end-to-end reproductions of those systems. Independently executed turn-final controls are labeled separately. We make comparative claims only within each stated interface and estimand.

Controlled execution contract. Each arm starts from the same reset database and receives the same waveform, tool schema, callback payload, and event clock. The harness records dispatch, policy output, gate verdict, effect, and final state. Trace–snapshot disagreement, timeouts, crashes, and missing traces remain failures. This design prevents confounding from different results, initial states, or selective exclusion. The comparison measures authorization under controlled streaming conditions, not general conversational quality.

6.2 METRICS AND STATISTICAL PROTOCOL

The co-primary endpoints are wrong-commit rate and safe task completion. Safe task completion requires the task predicate to hold and zero wrong commits in the episode. This joint definition prevents a controller from obtaining a high safety score by refusing every effect. Secondary endpoints include stale, wrong-target, duplicate, and missed effects, as well as correction-to-cancel, callback-to-decision, and end-to-end latency.

The 516 synthetic-OOD families are the inferential units; four frozen seeds and two timings yield $516 \times 4 \times 2 = 4{,}128$ attempts per arm. We use 10,000 family-bootstrap replicates and two-sided family sign flips, and reject incomplete panels. Because the same benchmark family informed development, these tests measure frozen-OOD stability rather than independent post-selection confirmation. Natural-speech analyses likewise cluster by dialogue family; Appendix A specifies their denominators and controls.

Wrong commit denotes an unauthorized durable mutation. A stale effect is the temporal subclass: valid at dispatch but unauthorized at commit. A missed effect is an authorized mutation absent from the final state; a duplicate repeats the same logical request. Reporting safety with completion prevents blanket refusal from appearing successful. Resampling whole families preserves dependence across callback timings and policies instead of treating 4,128 attempts as independent.

6.3 RQ1: TRANSACTION SAFETY ON CONTROLLED SYNTHETIC SPEECH

Table 1 reports the controlled synthetic OOD comparison over 516 families and four policy seeds. The two arms use identical scenarios, tools, and callback clocks. DUPLEXTXN raises safe task completion from 8.6% to 82.6% and reduces unsafe-effect episodes from 17.1% to below 0.1% (one episode). All family-paired differences except duplicate effects have $p = 0.0001$ under the frozen sign-flip test; duplicate effects are too rare for a significance claim. Every attempt remains in the denominator: baseline counts for safe completion, missed effect, unsafe effect, and model failure are 354, 3,648, 704, and 50, versus 3,411, 732, 1, and 0 for DUPLEXTXN; categories may overlap. Because this synthetic panel informed development, it establishes controlled mechanism behavior rather than generalization to natural human speech.

The gain cannot be explained by indiscriminate blocking: DUPLEXTXN adds 3,057 safe completions while removing 703 of 704 unsafe episodes and 2,916 misses. In each pair, the callback should commit before the revision but be held or dropped after it. Always-commit and always-drop policies therefore each fail one member; success requires a decision that follows the evolving spoken mandate.

The outcome categories intentionally overlap. For example, an episode can both externalize an obsolete effect and miss the currently authorized one, so error counts need not sum to 4,128. Model fail-

ures are likewise retained in the common denominator rather than silently removed. This final-state accounting is stricter than tool-call accuracy alone: a syntactically correct call remains unauthorized when its authorizing intent has expired before the effect is committed.

Table 1: Controlled synthetic OOD diagnostic over 4,128 attempted episodes per arm (516 families $\times$ four seeds $\times$ two timings). Parentheses give counts. Arrows mark the preferred direction. Δ and 95% paired family-bootstrap CIs are DUPLEXTXN minus our execution of released DuplexOmni; percentage metrics are reported to one decimal, with rare nonzero events shown by inequalities rather than rounded to zero.

Metric	Released DuplexOmni (our run)	DUPLEXTXN	Δ (Txn$-$Omni)	95% CI
Safe completion ↑ (%)	8.6 (354)	**82.6 (3,411)**	+74.1	[70.5, 77.5]
Unsafe episodes ↓ (%)	17.1 (704)	$<$ 0.1 **(1)**	-17.0	[-19.4, -14.7]
Missed effects ↓ (%)	88.4 (3,648)	**17.7 (732)**	-70.6	[-74.2, -67.1]
Stale / episode ↓	0.072 (297)	**0 (0)**	-0.072	[-0.088, -0.056]
Duplicate / episode ↓	$<$ 0.001 (1)	**0 (0)**	-2.4×10^{-4}	[-7.3×10^{-4}, 0]

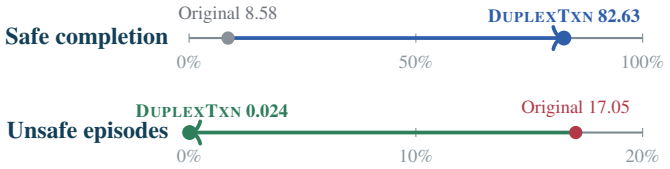

Figure 3: Headline synthetic-OOD results over 4,128 attempts per arm. Arrows point from Released DuplexOmni (our run) to DUPLEXTXN; Table 1 gives counts and paired intervals.

RQ4–RQ5: Post-freeze natural-speech transfer and strong controls. On 150 outcome-blind SpokenWOZ dialogue revisions, frozen DUPLEXTXN preserves all 600 authorized pre-revision commits, drops all 600 stale callbacks, and reaches 100% safe completion versus the observation-only baseline's 50% ($p < 10^{-4}$). In the harder 149-family native-call diagnostic, DUPLEXTXN raises safe completion from 8.72% to 19.30% (95% CI for the change [+5.70, +16.11]) and lowers stale-effect episodes from 24.16% to 6.54% ([-23.83, -11.58]), while missed commits increase by 0.67 percentage points. DUPLEXTXN does not dominate offline controls: latest-call wins reaches 29.53% completion but cannot decide before turn end and leaves 9.40% stale episodes. Appendix A reports the full natural-speech panels.

Controlled replay changes acoustics while fixing calls, isolating transfer of temporal authorization. Native calls additionally expose recognition, tool selection, and argument errors, explaining the smaller end-to-end gain. The turn-final control sees the completed utterance, so it is a utility reference, not a latency-matched streaming baseline. These results support controlled acoustic transfer but do not establish open-world full-duplex performance.

6.4 RQ2: WHY COMPARE DISPATCH AND COMMIT STATES?

The first architecture study holds the synthetic development split, callback clock, and executor fixed while changing only the callback representation. The commit-time-only head sees the current state; the paired head also sees the dispatch state, their difference, and element-wise product. Pairing lowers wrong commits and stale effects by about 88% and raises safe completion by 12.91 points (Table 2), supporting the observability argument in Section 3. This development experiment uses seeds 505/606/707/808 and is not an independent confirmation.

Pairing leaves callback-to-decision P95 unchanged at 1000.5 ms, so the gain does not come from waiting longer. Current state represents the current mandate; dispatch state identifies the earlier mandate that produced the callback. Their pair makes continued authorization observable.

6.5 RQ3: CAN THE POLICY OUTPUT BE SIMPLIFIED?

Post-freeze controls retain the causal encoder, paired representation, and gate while retraining only the output. Operation-only obtains zero stale effects by missing 97.46% of authorized commits;

Table 2: Callback-representation ablation, four development-seed means. Stale is the per-seed count; P95 is callback-to-decision latency.

Method	Wrong (%)	Missed (%)	Safe (%)	Stale/seed	P95 (ms)
Commit-time only	3.0102	20.9691	71.0514	44.25	1000.5
Paired state	**0.3571**	**13.0904**	**83.9632**	**5.25**	1000.5

intent-delta plus operation recovers utility but remains below joint state (Table 3). Matched linear, MLP, GRU, and Transformer controls attribute the main gain to temporal authorization and pairing rather than the Transformer block alone.

Table 3: Output-factorization ablation, four-seed means. Variants share the frozen encoder, paired representation, gate, clock, and episodes; only the output is retrained.

Output policy	Wrong (%)	Missed (%)	Safe (%)	Stale/seed
Joint transaction state	0.0170	**14.9023**	**82.6308**	0
Intent delta + operation	**0**	24.3078	71.5359	0
Operation only	**0**	97.4552	9.6657	0

Operation-only attains zero stale effects through near-universal refusal; intent delta restores utility but still permits incompatible factors. Joint-state supervision instead couples intent evolution to the action being externalized, distinguishing an obsolete callback from one that is valid but not yet ready.

Together, the two ablations separate information availability from output parameterization. The paired-state comparison changes the callback observation while fixing the executor and data split; the output-factorization study fixes the representation and gate while changing supervision. The observed gains therefore cannot be attributed solely to deterministic enforcement or Transformer capacity. Because both studies use development-derived synthetic data, they establish mechanism consistency rather than independent generalization.

7 Discussion

Current state alone cannot recover which earlier mandate authorized a callback; dispatch–commit pairing makes that relation observable and versioned runtime state enforces it. The operation-only output obtains nominal safety primarily by refusing authorized work, while matched MLP/GRU controls indicate that the Transformer block alone does not explain the improvement. Natural-speech and MiniCPM-o results support controlled acoustic transfer and representation portability, not open-world or cross-backbone superiority (Appendices A–B).

8 Conclusion

DuplexTxn reframes asynchronous full-duplex tool use as temporal authorization, combining learned intent comparison with deterministic commit enforcement. It reduces stale effects in controlled replay; evaluation in live deployment remains necessary.

AI use statement

Generative AI tools were used to propose and refine research questions, hypotheses, conceptual framing, methodology, and experiment design, assist software implementation and testing, and help construct and reformat synthetic training and diagnostic data. They were also used for literature discovery and synthesis, citation and reference formatting, bilingual wording, and drafting and editing portions of the manuscript and documentation. They assisted formulation and checking of the formal safety property and proof sketch, as well as interpretation of completed synthetic and external-panel results. They do not generate, impute, or replace empirical results. The authors review all AI-assisted work, verify literature claims against primary sources, execute deterministic and statistical

checks over the code and evidence, and take responsibility for the final text, claims, code, data, and artifacts.

ETHICS STATEMENT

This work studies agents that can trigger externally visible effects and therefore treats stale, misbound, unconfirmed, and duplicate actions as safety risks rather than only task errors. The experiments reported here use synthetic speech, a publicly released research corpus of human-to-human speech, and resettable test environments. No newly collected participant audio contributes to the reported model-quality results. SpokenWOZ is used under its research license, and we make no claims about speaker identity or demographic coverage. Deployment with real accounts additionally requires service-specific authorization, audit, recovery, and privacy controls beyond the studied gate.

REPRODUCIBILITY STATEMENT

The supplement specifies frozen model and source revisions, synthetic data construction, causal state capture, seeds, hyperparameters, baselines, environment adapters, denominators, and statistical procedures. Executable gates rebuild metrics from content-hashed runtime and environment evidence, reject incomplete assigned sessions, and independently reconstruct the frozen family–seed panel. The anonymized artifact excludes private data and infrastructure; two clean builds are byte-identical, all 299 content-bound members verify, 150 artifact tests pass, and a separate checker finds no violation in 1,536 gate states.

REFERENCES

Tanay Agrawal, Abid Ali, Antitza Dantcheva, and Francois Bremond. Scaling action detection: AdaTAD++ with transformer-enhanced temporal-spatial adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 12222–12231, 2025. URL https://openaccess.thecvf.com/content/ICCV2025/html/Agrawal_Scaling_Action_Detection_AdaTAD_with_Transformer-Enhanced_Temporal-Spatial_Adaptation_ICCV_2025_paper.html.

Mayur Akewar and Ravi Ranjan. SafeCommit: Certifying when memory-grounded agents may safely act. *arXiv preprint arXiv:2608.04289*, 2026. URL https://arxiv.org/abs/2608.04289.

Zheng Chen, Hanqing Liu, Duling Xu, et al. Cordon: Semantic transactions for tool-using llm agents. *arXiv preprint arXiv:2606.17573*, 2026. URL https://arxiv.org/abs/2606.17573.

Junbo Cui, Bokai Xu, Chongyi Wang, et al. Minicpm-o 4.5: Towards real-time full-duplex omnimodal interaction. *arXiv preprint arXiv:2604.27393*, 2026. URL https://arxiv.org/abs/2604.27393.

Alexandre Défossez, Laurent Mazaré, Manu Orsini, Amélie Royer, Patrick Pérez, Hervé Jégou, Edouard Grave, and Neil Zeghidour. Moshi: A speech-text foundation model for real-time dialogue. *arXiv preprint arXiv:2410.00037*, 2024. URL https://arxiv.org/abs/2410.00037.

Cristóbal Eyzaguirre, Eric Tang, Shyamal Buch, Adrien Gaidon, Jiajun Wu, and Juan Carlos Niebles. Streaming detection of queried event start. In *Advances in Neural Information Processing Systems 37, Datasets and Benchmarks Track*, pp. 100698–100733, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/b689b90ddf2b47d3103decabe6d47446-Paper-Datasets_and_Benchmarks_Track.pdf.

Guangyu Feng, Huanzhi Mao, Prabal Dutta, and Joseph E. Gonzalez. Concurrency without model changes: Future-based asynchronous function calling for llms. *arXiv preprint arXiv:2605.15077*, 2026. URL https://arxiv.org/abs/2605.15077.

Jun He and Deying Yu. Beyond memory: A transactional continuity kernel for long-lived AI agents. *arXiv preprint arXiv:2608.11632*, 2026. URL https://arxiv.org/abs/2608.11632.

Coleman Hooper, Minwoo Kang, Suhong Moon, Nicholas Lee, Eric Wen, John Wawrzynek, Michael W. Mahoney, Yakun Sophia Shao, Amir Gholami, and Kurt Keutzer. Speculative interaction agents: Building real-time agents with asynchronous I/O and speculative tool calling. *arXiv preprint arXiv:2605.13360*, 2026. URL https://arxiv.org/abs/2605.13360.

Ke Hu, Slyne Deng, Chen Chen, Elena Rastorgueva, Edresson Casanova, Punit Kumar, Dharmendra Choudhary, Nikhil Srihari, Ameya Sunil Mahabaleshwarkar, Viet Anh Trinh, Slim Essid, Oluwatobi Olabiyi, and Zhehuai Chen. A frontend-backend architecture for tool calls in full-duplex speech models. *arXiv preprint arXiv:2609.19334*, 2026. URL https://arxiv.org/abs/2609.19334.

Muye Huang, Lingling Zhang, Xingyu Yu, et al. Duplexomni: Real-time listening, seeing, thinking, and speaking for full-duplex interaction. *arXiv preprint arXiv:2606.09186*, 2026. URL https://arxiv.org/abs/2606.09186.

Xiaoyang Li, Yiqi Wang, Haohui Lu, Zhi Chen, Mo Li, Pingan Song, Mingkai Zheng, and Taotao Cai. MemTX: Transactional belief commit for stateful agent memory. *arXiv preprint arXiv:2607.23929*, 2026. URL https://arxiv.org/abs/2607.23929.

Guan-Ting Lin, Chen Chen, Zhehuai Chen, and Hung-yi Lee. Full-duplex-bench-v3: Benchmarking tool use for full-duplex voice agents under real-world disfluency. *arXiv preprint arXiv:2604.04847*, 2026. URL https://arxiv.org/abs/2604.04847.

Xudong Lu, Xueying Li, Annan Wang, Yang Bo, Jinpeng Chen, Zengliang Li, Nianzu Yang, Rui Liu, Xue Yang, Jingwen Hou, and Hongsheng Li. Omniinteract: Benchmarking real-world streaming interaction for real-time omnimodal assistants. *arXiv preprint arXiv:2605.26485*, 2026. URL https://arxiv.org/abs/2605.26485.

Bardia Mohammadi, Lars Klein, Akhil Arora, and Laurent Bindschaedler. Ghost tool calls: Issue-time privacy for speculative agent tools. *arXiv preprint arXiv:2606.02483*, 2026a. URL https://arxiv.org/abs/2606.02483.

Bardia Mohammadi, Nearchos Potamitis, Lars Klein, Akhil Arora, and Laurent Bindschaedler. Atomix: Timely, transactional tool use for reliable agentic workflows. *arXiv preprint arXiv:2602.14849*, 2026b. URL https://arxiv.org/abs/2602.14849.

Kensuke Nakamura, Lasse Peters, and Andrea Bajcsy. Generalizing safety beyond collision-avoidance via latent-space reachability analysis. In *Proceedings of Robotics: Science and Systems*, 2025. doi: 10.15607/RSS.2025.XXI.113. URL https://www.roboticsproceedings.org/rss21/p113.html.

Manik Rana, Calissa Man, Anotida Expected Msiiwa, Jeffrey Paine, Kevin Zhu, Sunishchal Dev, Vasu Sharma, and Ahan M. R. Agentchangebench: A multi-dimensional evaluation framework for goal-shift robustness in conversational ai. *arXiv preprint arXiv:2510.18170*, 2025. URL https://arxiv.org/abs/2510.18170.

Soham Ray, Keshav Dhandhania, Victor Barres, and Karthik R. Narasimhan. τ-voice: Benchmarking full-duplex voice agents on real-world domains. *arXiv preprint arXiv:2603.13686*, 2026. URL https://arxiv.org/abs/2603.13686.

Jesus Salas. Correct is not governed: Provenance integrity in agentic workflows. *arXiv preprint arXiv:2608.12761*, 2026. URL https://arxiv.org/abs/2608.12761.

Igor Santos-Grueiro. Temporary authority, permanent effects: Commit-time authorization for llm agents. *arXiv preprint arXiv:2607.10487*, 2026. URL https://arxiv.org/abs/2607.10487.

Kou Shi, Ziao Zhang, Shiting Huang, et al. Asynctool: Evaluating the asynchronous function calling capability under multi-task scenarios. *arXiv preprint arXiv:2605.27995*, 2026. URL https://arxiv.org/abs/2605.27995.

Bandhav Veluri, Benjamin N. Peloquin, Bokai Yu, Hongyu Gong, and Shyamnath Gollakota. Beyond turn-based interfaces: Synchronous LLMs as full-duplex dialogue agents. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 21390–21402. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.emnlp-main.1192. URL https://aclanthology.org/2024.emnlp-main.1192/.

Chengyou Wang, Hongfei Xue, Guojian Li, Zhixian Zhao, Shuiyuan Wang, Shuai Wang, Xin Xu, Hui Bu, and Lei Xie. Full-duplex interaction in spoken dialogue systems: A comprehensive study from the icassp 2026 humdial challenge. *arXiv preprint arXiv:2604.21406*, 2026a. URL https://arxiv.org/abs/2604.21406.

Siyin Wang, Wenyi Yu, Xianzhao Chen, Xiaohai Tian, Jun Zhang, Lu Lu, Yuxuan Wang, and Chao Zhang. End-to-end listen, look, speak and act. In *International Conference on Learning Representations*, 2026b. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/6d8af672c0bca7a4be6f0cd0d229c05c-Abstract-Conference.html.

Xiong Wang, Yangze Li, Chaoyou Fu, Yike Zhang, Yunhang Shen, Lei Xie, Ke Li, Xing Sun, and Long Ma. Freeze-omni: A smart and low latency speech-to-speech dialogue model with frozen LLM. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 63345–63354. PMLR, 2025. URL https://proceedings.mlr.press/v267/wang25aw.html.

Donghang Wu, Haoyang Zhang, Chen Chen, Tianyu Zhang, Fei Tian, Xuerui Yang, Gang Yu, Hexin Liu, Nana Hou, Yuchen Hu, and Eng Siong Chng. Chronological thinking in full-duplex spoken dialogue language models. In *Proceedings of the 27th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pp. 473–485. Association for Computational Linguistics, 2026. URL https://aclanthology.org/2026.sigdial-1.33/.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains. *arXiv preprint arXiv:2406.12045*, 2024. URL https://arxiv.org/abs/2406.12045.

Haoyang Zhang, Jun Chen, Donghang Wu, et al. Duplexsla: A full-duplex spoken language model with synchronized speech, language, and action. *arXiv preprint arXiv:2605.20755*, 2026. URL https://arxiv.org/abs/2605.20755.

Qinglin Zhang, Luyao Cheng, Chong Deng, Qian Chen, Wen Wang, Siqi Zheng, Jiaqing Liu, Hai Yu, Chao-Hong Tan, Zhihao Du, and Shiliang Zhang. Omniflatten: An end-to-end GPT model for seamless voice conversation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14570–14580. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.acl-long.709. URL https://aclanthology.org/2025.acl-long.709/.

Ruixiang Zhao, Hualei Wang, Renhe Sun, Enzhi Zhou, Jincenzi Wu, Xujie Song, Kexin Shi, Zihang Liu, Pengcheng Zhu, Jiayi Zhou, et al. Realtime-venus: A full-duplex interaction system with asynchronous delegation. *arXiv preprint arXiv:2609.13814*, 2026. URL https://arxiv.org/abs/2609.13814.

Yusheng Zheng, Yiwei Yang, Wei Zhang, and Andi Quinn. ACRFence: Preventing semantic rollback attacks in agent checkpoint-restore. *arXiv preprint arXiv:2603.20625*, 2026. URL https://arxiv.org/abs/2603.20625.

A POST-FREEZE SPOKENWOZ STRESS TEST

SpokenWOZ contains natural human-to-human task-oriented dialogue. Outcome-blind selection used validation-dialogue metadata and revision transcripts only. For each of 150 dialogue families, the harness extracts the causal audio prefix ending after a natural slot-value revision and replays the same already-issued old-value callback at two times: before the revision boundary, when committing is authorized, and after the complete 480-ms chunk containing the revision end, when the old result

must be dropped. The waveform, payload, model checkpoint, and execution clock are paired across arms. Four frozen policy seeds yield 1,200 evaluation instances per arm. No external example is used for training, selection, or tuning. This tests natural acoustics and spontaneous revision language under a controlled clock; the tool call is pre-initialized before the clip, so it is not a native overlapping full-duplex evaluation.

Table 4: Post-freeze SpokenWOZ panel. Counts aggregate 150 families, two callback timings, and four frozen seeds.

Metric	Observation-only	DUPLEXTXN	Difference
Safe completion (%)	50.0 (600)	**100.0 (1,200)**	+50.0
Wrong commit (%)	50.0 (600)	**0 (0)**	−50.0
Stale effect (%)	50.0 (600)	**0 (0)**	−50.0
Missed authorized effect (%)	0 (0)	0 (0)	0

A.1 NATIVE MODEL-ORIGINATED CALLS

We additionally replay 149 held-out SpokenWOZ dialogue families through the frozen released DuplexOmni stack without pre-initializing a tool call. The released model must infer each tool call and argument from causal natural speech. The observation-only control and DUPLEXTXN share the exact waveform, native causal states, model-originated calls, arguments, and callback clock; only DUPLEXTXN applies the frozen causal temporal policy and deterministic gate. The development family used to qualify the pipeline is excluded. No item is used for training, threshold tuning, or candidate selection.

Table 5: Post-freeze model-originated-call diagnostic on 149 public natural-speech dialogue families. Rates use dialogue family as the statistical unit; DUPLEXTXN averages the four frozen policy seeds. Intervals are 95% family-clustered bootstrap CIs for DUPLEXTXN minus the observation-only control. This supporting diagnostic does not replace the controlled synthetic main experiment.

Metric	Observation-only (%)	DUPLEXTXN (%)	Δ points	95% CI
Safe completion	8.72	**19.30**	+10.57	[+5.70, +16.11]
Wrong commit	70.47	**54.70**	−15.77	[−21.64, −10.40]
Missed commit	**70.47**	71.14	+0.67	[+0.17, +1.34]
Stale-result side effect	24.16	**6.54**	−17.62	[−23.83, −11.58]
Duplicate side effect	0	0	0	[0, 0]

The observation-only decomposition confirms that performance is not attributable to silent infrastructure failure: all 149 released-stack runs complete and 145 produce non-empty user ASR. At family level, 31 produce no model-originated call, 75 contain an off-target argument, and 36 execute an old result after the revision; categories overlap. Independent readback verifies one shared frozen released-model identity, S2 agent and seed, dispatch adapter, and 480-ms causal clock. This panel tests public natural acoustics plus native call generation, but remains offline replay with a controlled tool and does not establish unconstrained deployment performance.

A.2 TURN-FINAL STRONG-BASELINE PANEL

After freezing the native-call result, we registered four additional controls on the identical 149 families. Latest-call wins retains only the chronologically last completed call. Lexical revalidation retains the latest candidate whose value appears in normalized final ASR. A frozen Qwen3-8B selector reads the complete transcript and sanitized candidate calls, then commits at most one. A separate Qwen3-8B replanner reads the complete transcript and public tool schema and emits at most one replacement value. The prompts, deterministic decoding, model identity, retry budget, and fail-closed parse rule were frozen before any R656 decision. None of the four controls receives old/new labels, revision timestamps, baseline scores, or DUPLEXTXN outcomes; all 596 rows were independently reconstructed and all 267 model requests produced schema-valid JSON on the first attempt.

The panel provides evidence against attributing every gain solely to access to the final transcript: the replanner is conservative but misses most effects, and the selector still leaves more stale episodes

Table 6: Post-freeze turn-final controls on the same 149 native-call families. All rows are percentages; duplicate effects are zero for every method. DUPLEXTXN uses the four frozen policy seeds, whereas the other controls are deterministic. Turn-final controls cannot decide before the complete utterance is available. Bold marks the best observed value per metric, not a global winner.

Method	Safe ↑	Wrong ↓	Missed ↓	Stale ↓
Author observation-only	8.72	70.47	**70.47**	24.16
Latest call wins	**29.53**	49.66	**70.47**	9.40
Lexical revalidation	28.19	46.98	71.81	8.72
Turn-final LLM selector	28.86	47.65	71.14	10.07
Turn-final LLM replanner	18.12	**22.82**	81.88	10.07
DUPLEXTXN	19.30	54.70	71.14	**6.54**

than DUPLEXTXN. It also reveals an important trade-off. On this difficult panel, waiting until turn end and choosing the latest call yields higher completion than DUPLEXTXN. DUPLEXTXN's observed advantage is the lowest stale-effect rate under a causal prefix-only decision rule; this panel does not establish an end-to-end latency advantage.

The comparison is supporting evidence, not a confirmatory replacement for Table 1.

B SUPPORTING MINICPM-O CROSS-BACKBONE DIAGNOSTIC

Before opening these outcomes, we froze a readback of four already-produced MiniCPM-o 4.5 policy runs (seeds 911/1013/1217/1429). The replication uses the same two-layer causal-policy and paired-adapter design, but trains it from MiniCPM-o native streaming hidden states on the original synthetic splits. An independent job verified the six content-bound source files, the four seed identities, the result hash, and the non-main-table scope. Table 7 reports every seed through its mean and range. The 86.94% mean safe-completion rate indicates that MiniCPM-o states contain usable transaction signals. At the same time, the larger wrong-commit rate and nonzero stale-effect count expose a backbone-dependent safety calibration gap. Because this diagnostic has no unmodified MiniCPM-o baseline, reuses the synthetic OOD families, and does not evaluate public speech or native tool selection, we do not use it to claim cross-backbone superiority.

Table 7: Post-freeze supporting MiniCPM-o synthetic-OOD diagnostic. Values are four-seed means; brackets give the minimum and maximum seed values. Stale and duplicate effects are counts per seed.

Metric	Mean	Seed range
Safe task completion (%)	86.9428	[84.9806, 88.4690]
Wrong commit (%)	0.8333	[0.5442, 0.9524]
Missed commit (%)	13.8436	[12.1336, 18.4853]
Stale effects / seed	11.5	[8, 14]
Duplicate effects / seed	0	[0, 0]
Callback-to-decision P95 (ms)	1000.75	[1000, 1001]

C OUTPUT-FACTORIZATION ABLATION

Both post-freeze controls keep the final causal encoder, shared trunk, paired dispatch/current representation, Pair-MLP, deterministic gate, 1,032 synthetic OOD reports, event clock, and seeds 911/1013/1217/1429. Only the output is retrained. A five-operation head predicts no-op, call, wait, commit, or drop; a compositional head additionally predicts three intent deltas.

Table 3 reports the complete output-factorization comparison in the main paper; this appendix records its frozen controls and training scope.

A separate matched audit compares current-chunk linear and MLP heads, a causal GRU, and a causal Transformer. The Transformer improves safe completion by 3.29 points over the same-width linear head, is 0.58 points below the same-width MLP, and does not differ significantly from parameter-matched MLP or GRU controls on the primary endpoints.

D SHARED-SEMANTIC RUNTIME DIAGNOSTIC

We replay six local runtime controls on the frozen 4,128-observation panel. Unlike the end-to-end comparison, every control receives the same causal semantic events from the final controller; evaluator annotations appear only after decisions for scoring. This shared-stream estimand isolates executor behavior, not each cited system's perception or planning quality.

Table 8 shows that intent-version tracking alone is insufficient, whereas the stronger lineage, binding, confirmation, and idempotency checks in our Cordon- and Atomix-inspired controls recover a substantial portion of the joint safety–utility performance. DUPLEXTXN still improves safe completion over either transaction control by 10.30 points (paired family-bootstrap 95% CI [8.60, 12.04], two-sided family sign-flip $p = 0.0001$) and reduces unsafe episodes from 1.8653% to 0.0242%. This difference is consistent with the value of learned dispatch–commit semantic comparison beyond a fixed runtime gate, under the shared-semantic-stream estimand.

The shared-stream panel has no non-empty dependency edge, so our AsyncFC-inspired control's distinguishing future rule is not exercised; local synchronous blocking, naive asynchronous execution, and future handling consequently share endpoint values despite different interaction behavior. The two transaction variants also coincide because their distinguishing frontier condition is inactive. We do not impute model-reread or asynchronous-thinking controls whose required events are absent; Appendix A.2 instead reports independently executed turn-final controls without claiming author equivalence.

Table 8: Shared-stream runtime diagnostic on 4,128 frozen observations. Rows labeled "local" are our mechanism-level implementations, not official, end-to-end, or author-reported results. Stale and duplicate are per episode; other columns are episode percentages.

Runtime	Safe (%)	Unsafe (%)	Missed (%)	Stale/ep.	Duplicate/ep.
Local synchronous blocking	5.8140	94.1860	0	0.41667	0.93217
Local naive asynchronous	5.8140	94.1860	0	0.41667	0.93217
Local AsyncFC-inspired futures	5.8140	94.1860	0	0.41667	0.93217
Local goal-shift versioning	12.8876	70.7364	25.2907	0.01793	0.59981
Local Cordon-inspired transaction	72.3353	1.8653	26.4535	0.01793	**0**
Local Atomix-inspired transaction	72.3353	1.8653	26.4535	0.01793	**0**
DUPLEXTXN	**82.6308**	**0.0242**	**17.7326**	**0**	**0**

E SAFETY ARGUMENT AND CONFORMANCE CHECKING

Let L_t be the set of authorization epochs that are live at commit time t, and let a ticket be $q = (v_q, g_q, p_q, D_q, c_q, k_q)$: dispatch-time epoch, target, payload hash, dependency set, scoped-confirmation requirement, and idempotency key. For a candidate effect x returned under q, the reference gate evaluates

$$\begin{aligned} G_t(x, q) = {} & [v_q \in L_t] \wedge [\mathrm{known}(q)] \wedge [\mathrm{ready}(x, q)] \\ & \wedge [\mathrm{target}(x) = g_q] \wedge [\mathrm{hash}(\mathrm{payload}(x)) = p_q] \\ & \wedge [\forall d \in D_q : \mathrm{complete}(d)] \wedge [\mathrm{confirm}(c_q, v_q, t)] \wedge [k_q \notin C_t], \end{aligned} \tag{3}$$

where C_t is the committed-key set. A no-confirmation ticket makes the confirmation predicate vacuously true. The learned policy can request a commit, but the executor externalizes x only when that request and G_t are both true in one atomic check-and-record operation.

Ledger-relative safety. Assume tickets and lineage cannot be forged or rebound, all protected effects pass through the reference gate, and checking G_t, invoking an idempotent downstream operation, and recording k_q have atomic or transactional-outbox semantics. Then a candidate can commit only if every conjunct of G_t holds. In particular, a non-live epoch makes the first conjunct false independently of the learned policy score. Target, payload, dependency, confirmation, and duplicate-key violations fail their corresponding conjuncts. This is a direct runtime invariant over observable ledger state.

Semantic lifting condition. Let $A_t(q)$ mean that the user's current spoken mandate still authorizes the concrete effect bound to ticket q. Ledger-relative safety implies semantic no-stale-effect safety only under invalidation completeness: $\neg A_t(q) \Rightarrow v_q \notin L_t$ before commit. Under that condition, every semantically stale candidate fails the live-epoch conjunct. If the streaming policy misses a correction or cancellation, the old epoch can remain live and the deterministic gate alone cannot infer $\neg A_t(q)$. This is an end-to-end semantic failure, not a counterexample to the ledger invariant. Conversely, a false invalidation can reduce liveness or task completion without violating safety. Both error modes are therefore measured empirically. Neither claim guarantees that every valid effect eventually commits.

The independent finite-matrix checker enumerates live, revised, and revoked epochs together with every Boolean ticket-existence, result-readiness, callback-lineage, target/payload-binding, confirmation, dependency, and duplicate condition, and checks the implementation against G_t. Its report verifies ledger conformance under the runtime assumptions. It does not test the semantic lifting condition or prove that the policy correctly interprets speech. Those errors are measured separately by paired executable evaluation rather than hidden inside the deterministic safety claim.

F Training and implementation details

The base temporal policy is trained from scratch for each seed on 174,007 causal-prefix rows for 10 epochs with batch size 32, AdamW, learning rate 3×10^{-4}, weight decay 0.01, and dropout 0.1. Calibration uses 22,745 rows and selects checkpoints without gradient updates; the 26,711 synthetic-OOD rows are reserved for development diagnostics. The training rows are repeated causal prefixes derived from 3,976 training families rather than independent semantic examples. After freezing the temporal policy and its output heads, the paired adapter is trained for 40 epochs with batch size 64, AdamW, learning rate 3×10^{-4}, weight decay 0.01, and block-loss weight 0.5. Each of the four final seeds starts both stages from a new random initialization and reuses neither optimizer state nor a development checkpoint. Online evaluation accumulates only the causal prefix available at the current event boundary; an independent full-prefix replay matches its logits exactly on the frozen validation trace.

G Development and candidate freeze

Table 9 records the complete model-selection trajectory at the granularity required to distinguish development from confirmation. The early recurrent policy and all subsequent alternatives were evaluated only on synthetic training, calibration, or development-OOD partitions. Five factorizations of semantic validity and execution readiness were considered, along with three label-blind nested training fractions. A single 50% paired candidate was selected by a prospectively fixed lexicographic safety–utility rule and then retrained from random initialization under four previously unused seeds. Those four checkpoints were frozen before the final synthetic OOD replay. The later parameter-controlled structure and output-factorization studies report all tested alternatives but cannot replace the frozen candidate.

Table 9: Outcome access and decision role across model development. All rows use synthetic data; "Dev" denotes the development OOD partition.

Stage	Candidate set	Decision data	Role
Early policy	512-wide GRU with calibrated thresholds; seeds 17, 29, and 43	Dev	Diagnostic; superseded
Paired search	Five validity/readiness factorizations; nested 25%, 50%, and 100% training subsets; seeds 505, 606, 707, and 808	Training, calibration, and dev OOD	Select one recipe
Fresh freeze	50% paired candidate, retrained from scratch; seeds 911, 1013, 1217, and 1429	Dev OOD	Final evaluation checkpoints
Structure audit	Linear, MLP, GRU, and causal Transformer under common-width and parameter-matched controls	Dev OOD	Diagnostic only; no reselection

H RUNTIME AND EVALUATION IMPLEMENTATION DETAILS

The policy augments each native state with exactly six executor-observable facts: tool pending, result ready, dependency ready, assistant speaking, effect irreversible, and elapsed dispatch time. These facts contain neither the gold semantic relation nor the authorization label; in particular, gold intent-version distance and confirmation validity are not model inputs. The runtime instead checks its own version and scoped-confirmation records at commit.

Each audited waveform is converted once to 24 kHz mono PCM16 for content-hashed transport and divided into 480 ms chunks. The frozen released DuplexOmni service deterministically resamples submitted chunks to its 16 kHz neural frontend. A collector advances an isolated session only after the native state for the preceding chunk is durable. The capture audit verifies that future audio was absent from every prefix state. Online inference appends each new native state to a per-session causal prefix and re-evaluates that prefix; key/value caching is an implementation optimization and is not assumed by the reported latency.

For baseline comparability, a single online collector assigns a globally unique sequence number to model, tool, policy, and executor events. Every baseline consumes the identical normalized event stream, and tied timestamps are resolved by that sequence rather than by wall-clock order. These contract checks remain diagnostic and do not constitute benchmark results. The released-model control loads the clean frozen released DuplexOmni repository and checkpoint; its benchmark wrapper only supplies the fixed synthetic tool runtime and captures outcomes. It has no learned transaction policy, intent versioning, or deterministic gate. DUPLEXTXN alone binds the learned policy and activates the transaction gate. Aggregate scores are ignored during evidence promotion, and each commit decision is atomically bound to a separate executable-environment audit stream.

Privately collected human recordings and turn-final ASR controls are excluded from the empirical claims. The post-freeze SpokenWOZ stress test comprises the controlled and native-call diagnostics in Appendix A; both remain offline callback replay and are reported separately from the synthetic estimand. DuplexSLA, AsyncTool, and FDB-v3 therefore appear as related work rather than numerical rows: their released tasks, state interfaces, and outcome contracts do not instantiate the paired pre/post-revision intervention used here.

I COMPLETE SYNTHETIC STRUCTURE DIAGNOSTIC

All 30 prospectively registered structure–endpoint tests are retained. Against the same-width GRU, the causal Transformer changes safe completion by $+1.16$ points ($p_{\mathrm{Holm}} = 0.0506$), unsafe effects by -0.65 points, and missed authorized effects by -3.54 points (both $p_{\mathrm{Holm}} = 0.0030$). Against the same-width MLP, it changes safe completion by -0.58 points and missed effects by $+1.77$ points. Parameter-matched MLP and GRU controls do not differ significantly from the Transformer on the primary endpoints. These content-hashed, independently verified runs are architecture diagnostics and do not alter the frozen final candidate.

J MULTI-SESSION POLICY-RUNTIME DIAGNOSTIC

We prospectively fixed a policy-only diagnostic for one shared frozen final model with session-local causal prefixes and call bindings. Across all four frozen seeds, interleaving 1, 8, 32, or 64 sessions in round-robin chunk order produced decisions exactly identical to running every session in isolation; all prefix-storage and call-binding isolation checks passed. Mean per-session mutable state was 136 KiB. From one to 64 sessions, mean policy-operation P50 changed from 1.590 to 1.615 ms, P95 from 1.678 to 3.788 ms, and sequential round-robin throughput from 577.89 to 572.12 operations/s. These measurements exclude the audio encoder, Omni backbone, speech decoder, network, and tool service, so they establish state isolation and policy-layer overhead only, not full-service capacity or a confirmatory task-quality result.

K REPRODUCIBILITY CHECKLIST

Every run records commit, image, model hash, data-family split hash, seed, hardware, dependency versions, configuration, trace outputs, and checksums. Diagnostic-only episodes are rejected by the principal-results aggregator. The submission checker requires hash-bound evidence for backbone promotion, four-seed final policy training, incremental-runtime equivalence, complete paired released-DuplexOmni/DUPLEXTXN controlled replay, baseline reconstruction, ablation provenance, and the paired statistical report. A manuscript claim without the corresponding artifacts cannot pass the release check. The post-freeze SpokenWOZ evidence separately binds the official audio revision, the 150-family controlled panel, the 149-family native-call panel, extracted waveforms, native causal states, four frozen final-policy checkpoints, model-originated calls, paired statistics, four post-freeze turn-final controls, and immutable verification receipts. Mutable runtime logs are excluded from its integrity manifest. The exact bibliography bytes are also hash-bound to a per-entry primary-source verification manifest. Changing any reference invalidates the release check until its metadata is re-verified.

L PAIRED SYNTHETIC EVIDENCE PROMOTION AND RELEASE CHECKS

For every seed and frozen synthetic episode, our execution of released DuplexOmni and DUPLEXTXN operate in separate end-to-end runs. A fail-closed harness binds both arms to the same waveform, callback schedule, initial executable state, environment and adapter revisions, and frozen DuplexOmni checkpoint. It additionally attests that the released-model arm uses the clean frozen source tree and checkpoint without a learned transaction policy or transaction gate, whereas DUPLEXTXN binds the learned policy checkpoint and enables the transaction gate. Runtime oracle fields, noncausal logs, mock tool outcomes, and incomplete commit coverage are rejected.

The evidence builder ignores reported aggregate scores, cross-checks runtime decisions against the independently recorded environment audit, and reconstructs every row from content-hashed snapshots. A full-scale evidence pass requires all four frozen final seeds and all 516 synthetic-OOD families; reduced smoke fixtures are ineligible for Table 1. An independent readback reparses the per-decision rows, checks family identities and checksums, recomputes the principal point estimates, and reproduces the bootstrap intervals and permutation tests.

Events are durably recorded before export. The recorder assigns causal sequence numbers and monotonic timestamps and binds each commit decision to the post-transaction environment snapshot. Controlled replay uses deterministic decoding and one pre-materialized schedule of tool results and callback delays, rather than assuming that equal random seeds imply equal random draws. Each schedule names a zero-based tool-call ordinal and a nonnegative delay after the runner observes that call, so differences in internal call identifiers do not invalidate causal pairing.

M SYNTHETIC STATISTICAL PROTOCOL

Wrong-commit rate and safe task completion are joint principal endpoints so a controller cannot appear safe by refusing all effects. Whole scenario families are the resampling and permutation unit. We use 10,000 family-bootstrap replicates for confidence intervals and 10,000 fixed-seed family sign flips for paired p-values. Secondary endpoints are descriptive unless otherwise stated. Because the final architecture emerged from the same broader synthetic development process, these tests describe stability on the frozen OOD panel; they are not presented as a post-selection confirmatory test.

The analysis executable refuses incomplete family–seed panels: every family must contain all four final seeds in both arms, and the family identities must match the content-bound benchmark design. It also refuses to overwrite a previous result. Safe task completion is scored once per complete scenario and requires both the environment completion predicate and zero wrong commits anywhere in that scenario. Commit-error rates retain their decision-level denominators.

Before execution, every seed–episode assignment, synthetic waveform, callback schedule, and initial environment is frozen by content hash. Each arm binds that assignment-manifest hash at session start. Failed, timed-out, or incomplete sessions cannot be dropped or analyzed as complete cases. All four final seeds remain repeated measurements within each family; the 516 families, not the 2,064

seed–family observations, are the bootstrap and permutation units. The estimand is conditional on seeds 911/1013/1217/1429, and seed-level evidence is retained in the release package.