Typed Boundaries for Small-Model Web Agents

Tracking the Reference Agent Across Small Scales at a Third of the Tokens with No Invalid Actions Executed

Kyle W. Luecke
Driftbound Labs
ORCID: 0009-0008-0202-7852

September 2026

Abstract

Small language models deployed as browser agents fail in two characteristic ways: they emit actions that reference elements not on the page, and they repeat actions that change nothing. We describe a typed runtime boundary, SCP, that sits between a web page and a model: a typed projection of the accessibility tree replaces raw context, and a deterministic guard ladder checks every proposed action against the page's declared state before execution, returning a typed veto for one retry and abstaining, with provenance, on a second failure. Nothing in the boundary is learned.

We evaluate the boundary on MiniWoB++ under BrowserGym with 25 randomly drawn tasks, 10 seeds each, against the standard reference agent, a typed-JSON observation, and unassisted variants, using task-clustered bootstrap statistics as the primary test and tokenizer-exact token counts under a frozen metrics specification. On Qwen2.5 models across a 0.5B, 1.5B, and 3B parameter ladder, SCP with the gate tracks the reference agent on task success at each size (+0.8, +0.4, +0.8 pp; no detected difference from zero, with equivalence within ±10 pp TOST established only at 0.5B [90% percentile CI -2.4, +4.0 pp], at the margin / not established at 1.5B [-10.0, +10.0 pp], and unbounded at 3B [-14.4, +15.6 pp]) while using roughly a third of the observation tokens per step (3.1×, 3.8×, and 3.3× under exact BPE) and executing zero invalid actions, where the unassisted reference agent sent 16% to 45% of its actions to the browser as ungrounded or malformed; equivalence to typed JSON was supported within ±10 pp, while preregistered superiority over the raw tree was not supported (+6.0 pp, $p = 0.24$ vs alpha = 0.01).

The gate filters and does not repair: on a synthetic cohort (N = 300) it reduced committed policy violations from 152 to 0 at 58.7% coverage while task success was identical on every rung (101/300; replicating the repair null from the exploratory 56-world protocol), and on the browser its success effect was confined to loops a compact representation caused. On a next-generation base (Qwen3.5-2B) the free-text reference agent collapsed on form tasks through prompt-example over-imitation; a one-line repair recovered 1.2 pp (36.4% to 37.6%), while a typed observation with a typed action schema, unassisted, scored 52.4% with zero of 565 actions invalid, and the compact SCP projection with the gate scored 59.2% at 66.2 tokens per observation against 235.2 for typed JSON (a 3.55× reduction) and 212.6 for GenericAgent (a 3.21× reduction). An attribution arm isolating the action schema from the observation (ARM_AXTREE_JSON_ACTION) scored 60.0% unassisted (+22.4 pp over the repaired reference agent), showing that schema-adherent action formatting drives the task completion lift, while observation typing alone showed a directional, non-significant deficit (-7.6 pp, CI [-18.8, +4.8], $p = 0.20$). Containment is an interaction of base model schema fidelity and the gate: an older base leaked 243 actions under the same unassisted typed-JSON arm that a newer base ran clean (0/565). A distilled multimodal specialist (A3-2B) executed 588 actions with none invalid under its distinct coordinate-action grammar (upper bound 0.5%). On models that follow the schema, the gate issued zero vetoes and contributed no success change; its value is the formal tail guarantee and the audit receipt. Structure did not substitute for scale: a 1.5B model with the boundary remained 4.4 pp below an unassisted 3B. All experiments were preregistered with hashed criteria, run by coding agents under human-signed freezes; episode retries were recorded as zero where the harness counted them (the AXTree/JSON-action arm); the other Qwen3.5-2B arms did not record retry counts. The code and data are released with sealed task lists, seeds, receipts, and a replication from the execution commits is planned (the preregistration commit fixes the plan; the execution commit fixes the code).

Contributions

1. A typed observation and deterministic guard ladder for browser agents, with an explicit provenance definition of abstention (abstention with committed action null is distinct from a model-issued cancel).
2. No detected difference from the standard reference agent on task success across small scales (+0.8, +0.4, +0.8 pp; clustered p = 0.77, 0.91, 0.94), with equivalence within ±10 pp TOST established only at 0.5B (90% percentile CI [-2.4, +4.0] pp), at the margin / not established at 1.5B ([-10.0, +10.0] pp), and unbounded at 3B ([-14.4, +15.6] pp), while operating at roughly one third of observation tokens (3.1× to 3.8× exact BPE) and with zero invalid actions executed, on an external oracle with task-clustered statistics. On the confirmatory battery, equivalence to typed JSON was supported, but preregistered superiority over raw accessibility trees was not supported (+6.0 pp, p = 0.24 vs alpha = 0.01).
3. A mechanism result: the gate filters and does not repair, observed on a 300-world population cohort where task success was identically 101/300 on every rung (confirming the repair null from the 56-world exploratory protocol), and on the browser where the gate's success effect was confined to representation-induced loops. On models that natively adhere to the schema (ARM_D), the gate fired zero vetoes and produced zero success difference (131/250 vs 131/250).
4. An attribution ladder on a next-generation small base isolating action-schema typing from observation typing: action-schema adherence drives the task completion lift (+22.4 pp), while observation typing alone showed no task success benefit (-7.6 pp exploratory, CI [-18.8, +4.8], p = 0.20).
5. A channel comparison against a distilled multimodal specialist, showing where a runtime gate's job ends (grounding) and the model's begins (targeting), with explicit metric separation across action grammars.
6. The method: preregistration, sealed cohorts, external seeds, agent execution with human-signed freezes, disclosed interim looks and harness changes, and a replication table.

Preregistered Hypotheses and Verdicts

Each hypothesis block below is extracted verbatim by verification script against its frozen pre-commitment hash. Sidecars were first committed after the freezes (0572af3); verification rests on the preregistration's bytes at its freeze commit. Evaluated verdicts are agent-drafted, pending human sign-off.

Synthetic Cohort (Phase 3 Confirmatory, N = 300)

Source Document: `PREREGISTRATION_PHASE3_CONFIRMATORY_300.md` (Frozen Pre-commitment SHA-256: `9b4fb52e c5ec11eb652b2b8079c072e043d329872035a4d7bc5a910d68d2fa67`); Ethereum block `26,028,195`.

- **H1 (Harmful Action Elimination)**:

  *"Level `D4` deterministic assistance will achieve **0.0% committed harmful violations** (*$0/300$*, upper 95% CI* $< 1.5\%$*), compared to the unassisted baseline `D0` (*$> 30\%$ *committed violations)."* **Verdict**: **Supported**. Level D4 assistance achieved 0.0% committed harmful violations (0/300, 95% Wilson score interval [0.0%, 1.24%], strictly below the 1.5% ceiling), compared to 152/300 (50.7% > 30%) under unassisted baseline D0.

- **H2 (Precision Scaling under Provenance Standard)**:

  "Task precision on committed actions will monotonically increase across rungs:

  $$\textit{Precision}(D0) \leq \textit{Precision}(D1) \leq \textit{Precision}(D2) \leq \textit{Precision}(D3) \leq \textit{Precision}(D4)$$

  " **Verdict**: **Supported (by construction)**. Precision on committed actions scaled strictly monotonically across rungs: 33.7% (D0: 101/300) <= 33.7% (D1: 101/300) <= 47.0% (D2: 101/215) <= 51.0% (D3: 101/198) <= 57.4% (D4: 101/176). This monotonicity is the mechanical consequence of filtering: task successes were invariant at identically 101/300 on every rung, while the committed denominator progressively shrank as the gates vetoed violating proposals.

- **H3 (Matched Coverage Superiority)**:

  *"At every matched coverage level $C \in \{C_{D1}, C_{D2}, C_{D3}, C_{D4}\}$, the deterministic assistance rung will achieve:

- Higher task precision than the confidence-thresholded baseline ($\Delta\text{Prec} > 0$).
- Substantially fewer committed violations than the confidence-thresholded baseline ($\Delta\text{Viol} < 0$)."* **Verdict**: **Not informatively tested as preregistered**. Unassisted model outputs carried a constant confidence field (1.0), collapsing the prespecified baseline to an output-length filter. Under a separate preregistration evaluating length-normalized token log-probabilities (`phase3-true-confidence-prereg-v1`, Frozen Pre-commitment SHA-256: `d3ca65af373486d4c7b4a352a98aebcec8fad3d80ed28f9cfb318b78543affa2`), the baseline as run was scored with llama.cpp on Q6_K weights by regenerating completions rather than teacher-forcing the original D0 outputs; 16 of the 300 regenerated completions diverged from the original sealed actions (revealing greedy decoding nondeterminism across environments), rendering the earlier reported comparison (+8.0 pp precision, 0 vs 65 violations) unreliable until rescored. Under the preregistration-conformant teacher-forced rescoring in HuggingFace PyTorch FP16 on CPU (`replication-audit-rescoring-v1` Arm R3, commit `68ae0fb`; see Initial Results §6), the confidence baseline at matched D4 coverage (58.7%, 176/300, mean logprob cutoff ≥ -0.661) achieved 39.2% task precision (69/176) with 88 harmful violations (29.3%), compared to 57.4% precision (101/176, +18.2 pp) and 0 violations for D4.

- **Reported Empirical Finding (Repair Null)**: Task success was identically 101/300 across all active rungs, including unguided retry (`D0_RETRY`, 101/298) and bounded guard retries (`D1–D4`). Deterministic assistance filters invalid actions but never repairs failed tasks. (The formal repair contrast was prespecified in the exploratory 56-world protocol `phase3-ladder-v1` [Contrast 4 / H3-D] and also retained the null).

MiniWoB Battery (25 Tasks × 10 Seeds × 5 Arms, N = 1,250)

Source Document: `PREREGISTRATION_MINIWOB_CONFIRMATORY.md` (Frozen Pre-commitment SHA-256: `5b6c0a91512b16fa77435f923647b96c0b77e637f15a0f8190bc99f7dd691a41`); frozen harness commit `9607653`.

- **Hypothesis H1 (Per-Observation Representation Compression)**:

  *`SCP_STATIC` (Arms E and E_D4) will reduce model-visible context tokens per observation by: - $\geq$ **60**% relative to `TYPED_STRUCTURED` (Arms D and D_D4) - $\geq$ **50**% relative to `a11y_tree` (Arm A) while maintaining identical control entity coverage. Token counts will be evaluated per observation slice, strictly separated from behavioral step counts.* **Verdict**: **Supported**. Observation prompt tokens per step for SCP-D4 averaged 63.5 (exact BPE; 47 char proxy) vs 234.6 BPE / 197 char proxy for `TYPED_STRUCTURED` (72.9% BPE reduction / 3.69× compression; 76.1% char reduction) and 253.5 BPE / 174 char proxy for raw `a11y_tree` (75.0% BPE reduction / 3.99× compression; 73.0% char reduction), exceeding the >= 60% and >= 50% thresholds under both exact BPE and character proxy standards.

- **Hypothesis H2 (Assistance Universality & Representation Interaction)**:

  The D4 assistance ladder will significantly reduce invalid proposals and idempotent repetition across representations: 1. On `TYPED_STRUCTURED`: `ARM_D_D4` $\geq$ `ARM_D`. 2. On `SCP_STATIC`: `ARM_E_D4` $>$ `ARM_E`. The assistance gain ($\Delta_{assist} = Success_{D4} - Success_{D0}$) will be evaluated across both representations to determine whether deterministic assistance is universally additive or specifically compensates for compact syntax. **Verdict**: **Not supported on named outcomes (failed prediction)**.

  - **Invalid proposals**: Emitted invalid proposals were not reduced by the gate; retries increased the count (243 -> 329 on typed JSON, 119 -> 340 on SCP) because turn-2 retry prompts gave the model additional opportunities to propose actions. First-attempt invalid proposal rates per step were 33.8% (243/719) on typed JSON and 15.5% (119/769) on SCP, confirming that the gate filters invalid actions before browser execution rather than suppressing model proposal generation.
  - **Idempotent repetition**: Consecutive identical executed actions fell directionally across both representations, but neither reduction cleared statistical significance under the primary task-clustered bootstrap test: on typed JSON, repeats fell by 10 (-3.4%, 293 on `ARM_D` to 283 on `ARM_D_D4`; per-episode diff: -0.0400, cluster SE = 0.0466, 90% CI [-0.1200, +0.0240], 95% CI [-0.1400, +0.0400], clustered p = 0.4344); on SCP, repeats fell by 58 (-14.8%, 393 on `ARM_E` to 335 on `ARM_E_D4`; per-episode diff: -0.2320, cluster SE = 0.2295, 90% CI [-0.6240, +0.1320], 95% CI [-0.7040, +0.1921], clustered p = 0.3246).
  - **Task success**: Assistance gain was representation-specific and did not clear significance under the primary test: on typed JSON, `ARM_D_D4` >= `ARM_D` held weakly (+0.8 pp, 35.6% vs 34.8%, clustered difference p = 0.88, McNemar p = 0.48); on SCP, `ARM_E_D4` gained +6.0 pp over `ARM_E` (37.6% vs 31.6%), which does not clear the primary task-clustered significance threshold (p = 0.084 >= alpha = 0.05, cluster SE = 3.4%), though the

secondary paired McNemar test on matched task-seed episodes showed 15 paired wins to 0 (p = 0.0003). Deterministic assistance compensated for compact syntax rather than being universally additive.

- **Hypothesis H3 (Task Success Equivalence / Non-Inferiority)**:

  *ARM_E_D4 will demonstrate task success equivalence to ARM_D and ARM_D_D4 within a prespecified **Two-One-Sided-Tests (TOST) equivalence margin of** ± 10.0 **percentage points** ($\Delta \in [-10.0\ pp, +10.0\ pp]$ at $\alpha = 0.05$), while achieving a statistically significant superiority over raw accessibility trees (ARM_A, $p < 0.01$).* **Verdict**: **Mixed**.

  - **Equivalence to typed JSON**: **Supported**. ARM_E_D4 vs ARM_D is +2.8 pp (90% clustered CI [+0.0, +6.0] pp, 95% CI [-0.4, +6.8] pp) and vs ARM_D_D4 is +2.0 pp (90% clustered bootstrap percentile CI [-0.40, +5.20] pp, 95% CI [-0.80, +6.00] pp), both strictly inside the +-10.0 pp TOST margin.
  - **Superiority over raw accessibility trees (ARM_A)**: **Not supported** at p < 0.01. ARM_E_D4 vs ARM_A is +6.0 pp (95% CI [-3.6, +17.2] pp, clustered difference p = 0.24 >= 0.01).

- **Hypothesis H4 (Deterministic Safety & Zero Execution Violations)**:

  *Under both assisted arms (ARM_E_D4 and ARM_D_D4), the deterministic assistance ladder will achieve **0 invalid browser interaction proposals reaching Playwright execution** (100% interception of ungrounded bids, disabled elements, and redundant state transitions).* **Verdict**: **Supported by construction**. Both ARM_E_D4 and ARM_D_D4 executed zero invalid actions across all 250 episodes (0/716 and 0/731 attempted steps). The substantive empirical measure is leaks without the gate (ARM_D: 243 shadow detections, ARM_E: 119 shadow detections).

Reference Agent and Model Scaling Ladder

Source Document: PREREGISTRATION_EXTERNAL_BASELINES_AND_LADDER.md (Frozen Pre-commitment SHA-256: 293af04472c3395390254d8cc71b0214f4f72064dac0a0cf89ad7a70ad19a595).

- **Hypothesis H1 (Reference Baseline Parity)**:

  - ARM_E_D4 (Guarded SCP on Qwen2.5-1.5B) achieves parity or superiority relative to ARM_GENERIC_AGENT (GenericAgent on Qwen2.5-1.5B) while operating at $\geq 60\%$ fewer prompt observation tokens. **Verdict**:

  - **Token reduction clause**: **Supported**. Observation tokens per step reduced from 242.3 to 63.5 (73.8% reduction / 3.82× exact BPE; 3.55× char proxy), exceeding the >= 60% criterion.
  - **Task success parity / superiority clause**: **Not established under strict equivalence boundary**. Observed difference is +0.40 pp (37.6% vs 37.2%, difference-test clustered p = 0.9112). Under the primary task-clustered bootstrap (B = 10,000, default seed 42) with a +-10.0 pp equivalence margin, the 90% percentile CI is [-10.00, +10.00] pp. The lower bound lands at exactly -10.00 pp, failing non-inferiority under strict inequality (> -10.0 pp; 4 of 12 bootstrap seeds strictly clear -10.0 pp; weak inequality >= -10.0 pp holds across all 12 seeds).

- **Hypothesis H2 (Model Scale Replacement)**:

  - ARM_E_D4 with Qwen2.5-1.5B achieves equal or higher task success than unassisted Qwen2.5-3B (ARM_E on 3B). - "Deterministic structure and transition gating replaces a doubling of model parameters." Verdict**:** Unexecuted / Protocol Deviation**. Unassisted SCP on 3B (ARM_E on 3B) was not evaluated in the ladder sweep (the evaluated 3B arms were ARM_GENERIC_AGENT and ARM_E_D4). An exploratory comparison against unassisted 3B GenericAgent (42.0%) shows a -4.4 pp deficit (37.6% vs 42.0%, cluster SE = 7.7%, p = 0.58, 95% CI [-19.6, +10.8] pp). The preregistered comparator (ARM_E on 3B) is scheduled as a labeled late completion (Arm R4) under the frozen replication protocol (PREREGISTRATION_REPLICATION.md, SHA-256 b17d001f..., commit 9e40dde), to be evaluated after all other ladder results were known.

- **Hypothesis H3 (Ladder Universality Across Scales)**:

  - Deterministic assistance (D4) reduces runtime execution leaks to 0 across all model scales (0.5B, 1.5B, 2B, 3B). - The gate's positive effect is directional and concentrated in multi-step form and sequencing tasks across all model sizes. **Verdict**:

  - **Containment clause**: **Supported**. Executed invalid actions under D4 dropped to zero across all four evaluated rungs (0/694 at 0.5B, 0/716 at 1.5B, 0/540 at 2.0B, and 0/598 at 3.0B).

- **Task-concentration clause**: **Not testable as preregistered**. The isolated gate effect (E_D4 vs E) was only evaluated at 1.5B; comparing E_D4 vs GenericAgent across the ladder conflates observation representation with the gate. Task strata were sampling categories from the MiniWoB suite rather than prespecified ladder contrast tests.

Exploratory Analyses (Not Preregistered)

- **Scale Invariance Across Rungs (0.5B and 3B Parity)**: The preregistration prespecified baseline parity (H1) specifically on the 1.5B model. Testing equivalence across the remaining rungs is an exploratory extension (reported without a formal preregistered verdict):
  - At 0.5B: Observed difference is +0.80 pp (28.0% vs 27.2%, cluster SE = 1.85%, p = 0.7710); the 90% percentile bootstrap interval is [-2.40, +4.00] pp, falling strictly within the +-10.0 pp margin (non-inferiority supported post-hoc).
  - At 3.0B: Observed difference is +0.80 pp (42.8% vs 42.0%, cluster SE = 9.19%, p = 0.9384); the 90% percentile interval is [-14.40, +15.60] pp (leaving equivalence unbounded under the +-10.0 pp TOST margin), and the 95% CI is [-17.20, +18.80] pp (which was previously mislabeled as 90%).
  - Across all rungs (0.5B, 1.5B, 3.0B), difference from zero was null (+0.8, +0.4, +0.8 pp; clustered p = 0.77, 0.91, 0.94).
- **Per-Episode Token Ratios and Looping Dynamics**: While H1 specified per-observation compression, per-episode token consumption is modulated by model-specific looping: GenericAgent exhibited severe looping on 1.5B (5.23 steps/ep vs 2.86 for SCP-D4, yielding a 7.44× per-episode token economy ratio), while 0.5B and 3.0B looped less (2.94× and 3.34× per-episode token ratios).
- **Scale Substitution Gap**: 1.5B with SCP-D4 (37.6%) vs unassisted 3B GenericAgent (42.0%), a -4.4 pp deficit (p = 0.58, 95% CI [-19.6, +10.8] pp).
- **Qwen3.5 Attribution & Prompt Fragility (N = 250 per cell)**: Evaluated without a preregistration as an exploratory causal diagnosis:
  - **Prompt-example over-imitation**: The reference agent's collapse on forms was traced to single-argument fill examples in the prompt; adding a two-argument fill example recovered `enter-text` from 0/10 to 9/10, yielding a +1.2 pp battery gain (36.4% to 37.6%). A second grammar failure (unquoted click arguments) survived the prompt repair.
  - **Action-schema vs. observation typing**: Isolating action-schema typing on the raw accessibility tree (`ARM_AXTREE_JSON_ACTION`) achieved 60.0% unassisted (+22.4 pp over repaired GenericAgent), showing that schema-adherent action formatting drives the task completion lift. Observation typing alone (`ARM_D`, 52.4%) showed an exploratory, non-significant negative shift (-7.6 pp, CI [-18.8, +4.8], p = 0.20; see illustrative grounding case in Results §4). `ARM_D_D4` completed at 52.4% (131/250) with 0 vetoes, showing that the gate adds zero task success when the base model natively adheres to the schema.
  - **Containment interaction**: The unassisted typed-JSON arm leaked 0 actions on Qwen3.5-2B (0/565) vs 243 on Qwen2.5-1.5B (243/719), demonstrating that containment is an interaction of base model schema fidelity, observation grounding, and the runtime gate.
  - *Harness Audit Note*: Episode retries were recorded as zero where the harness counted them (the AXTree/JSON-action arm); the other Qwen3.5-2B arms did not record retry counts. The watchdog, environment-teardown timeout, retry logic, and per-episode checkpointing were added mid-sweep without a commit; the harness was preserved post hoc on 2026-09-24 (commit 1790bd3). Result files carry no per-episode code hash, so the generating code for the Qwen3.5-2B attribution runs is unverified.
- **A3 Multimodal Channel Comparison (Exploratory)**: A3-2B multimodal scored 52.0% unassisted (0 of 588 actions invalid, upper bound 0.5% under coordinate grammar; forms 10/10; click-link, click-checkboxes-large, book-flight 0/10). Grounding is solved by distillation while targeting is not; the gate's contribution on such a model is the guarantee and the receipt. Note: the result file for A3-2B, along with the exploratory 1.5B AXTree/JSON-action run, was committed post hoc (121c28b) with generating code unverified.

Method

1. The Boundary

SCP consists of three separable layers:

1. **Typing**: The environment's state is read into an intermediate representation (IR), and the model answers in a typed action schema ({action, target_bid, value}) rather than free text.
2. **Compression**: The IR is rendered either verbosely as typed JSON (ARM_D) or as SCP's compact algebraic projection (ARM_E); both are deterministic functions of the same IR.
3. **Gating (The Deterministic Assistance Ladder)**: Proposed actions are verified in fixed order before execution. The paper evaluates two domain instantiations of this hierarchy:
   - **Synthetic Cohort (Phase 3 Execution)**: Ran through the Python reference validator (deterministic_validator.py) with bitwise conformance verified against native OCaml binaries (ocaml_bridge.py implementing epistemic.ml, policy.ml, and transition.ml): D1 structural schema validity, D2 policy capability limits, D3 epistemic validity (vetoing actions based on variables marked Unknown, Unavailable, Ambiguous, or Stale), and D4 state transition invariants (vetoing actions lacking valid forward transitions, which caused D4 to abstain across the entire transition sector).
   - **Browser Execution (Phase 4 BrowserGym Adapter)**: Grounds the hierarchy to web accessibility semantics:
     - **D1 (Structural/Syntax)**: Validates JSON format, allowed action type (click, fill, select_option, noop), and presence of target_bid.
     - **D2 (Policy/Capability Guard)**: Verifies interactivity, rejecting actions targeting disabled elements (is_disabled).
     - **D3 (Epistemic Grounding)**: Confirms the target BID exists in the active observation's declared control map (the browser analogue to epistemic verification).
     - **D4 (Transition Invariants & Idempotence)**: Enforces role-action type compatibility (e.g., fill restricted to editable input roles) and checks for redundant state changes (vetoing fill proposals when the target field already contains the target string, and select_option when already selected).

A failed check returns a typed veto naming the offending rule and field; the model gets one retry; a second failure ends the episode as ABORTED_UNRESOLVED with committed action null. A model-issued cancel is a committed action and is never merged with abstention. Importantly, the browser D4 gate does not predict dynamic DOM state mutations resulting from navigation clicks; it strictly enforces role compatibility and suppresses idempotent repetitions.

Receipt: Per action: proposal, checks run, veto text if any, executed action, witness hash of the observation checked against.

2. Harness

BrowserGym with MiniWoB++ at commit 7fd85d7, headless Chromium under Playwright, greedy decoding ($\tau = 0.0$), pinned GGUF weights with SHA-256 verification, llama.cpp on a Vulkan backend or CPU as recorded per run. Models: Qwen2.5-Instruct at 0.5B, 1.5B, 3B (Q6_K); Qwen3.5-2B (reasoning off, recorded); A3-Qwen3.5-2B with its projector for the multimodal arm. The watchdog, environment-teardown timeout, retry logic, and per-episode checkpointing were added mid-sweep without a commit; the harness was preserved post hoc on 2026-09-24 (commit 1790bd3). Result files carry no per-episode code hash, so the generating code for the Qwen3.5-2B attribution runs is unverified.

3. Arms

- ARM_A: raw accessibility-tree text, unassisted (superseded as native baseline by GenericAgent; reported for continuity).
- ARM_GENERIC_AGENT: BrowserGym's GenericAgent, text-only configuration (use_ax_tree, no screenshot, no set-of-marks; flags recorded), the field's standard.
- ARM_GENERIC_AGENT_REPAIRED (Qwen3.5 only): reference prompt with a two-argument fill example; a disclosed repair.
- ARM_AXTREE_JSON_ACTION (Qwen3.5 only): reference observation with the typed JSON action schema; isolates action typing.
- ARM_D / ARM_D_D4: typed JSON observation, unassisted / gated.
- ARM_E / ARM_E_D4: SCP compact observation, unassisted / gated.
- ARM_A3_MULTIMODAL: A3-2B with screenshots; a distinct observation and action channel, labeled as such.

4. Tasks and Seeds

Synthetic cohort: 300 tasks with sector labels (routine, transition, refusal-required) and a sealed seed from a public Ethereum block hash. MiniWoB: 25 tasks drawn at random from the MiniWoB++ set with a recorded draw seed, 10 seeds (101–110) per task, sealed before the run; the 5-task pilot is disclosed and its tasks identified.

5. Statistics

Primary: task-clustered bootstrap ($B = 10,000$, resampling the 25 tasks) for every between-arm difference; Wilson intervals for rates; TOST equivalence with a stated ±10 pp margin, labeled generous. Secondary: McNemar on paired (task, seed) cells, never in the abstract. Metric definitions are frozen in METRICS.md (hash 911a930b…): T_{obs} is tokenizer tokens for the model that ran the episode ($T_{\text{obs, est}}$ character proxy is maintained as a labeled secondary column); leak rates carry explicit denominators.

6. Provenance and Process

Preregistrations hashed before runs; sidecars were first committed after the freezes (0572af3); verification rests on the preregistration's bytes at its freeze commit. Cited commits reside in a private development repository; the public repository carries a provenance manifest mapping each cited commit to the blob hashes of the files it depends on, and the full history is deposited as a restricted-access archive (DOI) whose SHA-256 was timestamped on 2026-09-28. Interim looks logged and disclosed; mid-run harness changes disclosed; errata written the day found; every close carries an artifact inventory with hashes; a replication from the execution commits is planned (the preregistration commit fixes the experimental plan; the execution commit fixes the running code), reported as a table with any disagreement and its cause. Existing experiments' freeze times rest on git commit blobs and Drive timestamps, and a retroactive OpenTimestamps backfill (`anchored_retroactively` on 2026-09-24) proves document state as of that date rather than the historical freeze; future freezes are anchored via OpenTimestamps directly by the Rule-19 freeze script prior to outcome-data access. Experiments were executed by coding agents; freezes, task selection rules, and arm definitions were human-signed. The human authors wrote the final prose.

Initial Results

1. Synthetic Cohort (N = 300, 6 arms, 1,800 cells)

Rung	Task Successes	Committed Harmful Violations	Coverage
D0 (unassisted)	101/300	152	100%
D0 + retry	101/300	—	—
D4 (full ladder)	101/300	0	58.7%

Routine sector: 50/75 on every arm, 0 violations. Transition sector: 0/45 on every arm; D4 abstained on all of it. Matched-coverage baseline: see erratum (length filter, not confidence).

2. MiniWoB Battery (25 tasks × 10 seeds × 5 arms; N = 1,250)

Arm	Success	95% Wilson CI	T_obs (Exact BPE)	T_obs (Char Est)	Executed Invalid Actions
ARM_A (raw tree)	31.6%	[26.2, 37.6]	253.5	174	331
ARM_D (typed JSON)	34.8%	[29.2, 40.9]	234.6	197	243
ARM_D_D4	35.6%	[29.9, 41.7]	229.9	193	0
ARM_E (SCP)	31.6%	[26.2, 37.6]	73.1	54	119
ARM_E_D4	37.6%	[31.8, 43.7]	63.5	47	0

Clustered comparisons (95% CI): E_D4 − D = +2.8 pp [-0.4, +6.8], p = 0.057; E_D4 − D_D4 = +2.0 [-0.8, +6.0]; E_D4 − A = +6.0 [-3.6, +17.2], p = 0.24; E_D4 − E = +6.0 [0.0, +15.2], p = 0.084 (15 paired wins to 0, in multi-step tasks). Equivalence to D within ±10 pp: supported (90% CI [0.0, +6.0]).

3. Reference Agent and Model Ladder (N = 250 per cell)

Model	GenericAgent	SCP-D4	Δ (Clustered SE)	T_obs (Exact BPE)	T_obs (Char Est)	Ratio (Exact BPE)	Ratio (Char Est)	Leaks Without Gate
Qwen2.5-0.5B	27.2%	28.0%	+0.8 (1.9)	203.7 vs 65.3	139.8 vs 48.3	3.12×	2.89×	314/706 (44.5%)

Model	GenericAgent	SCP-D4	Δ (Clustered SE)	T_obs (Exact BPE)	T_obs (Char Est)	Ratio (Exact BPE)	Ratio (Char Est)	Leaks Without Gate
Qwen2.5-1.5B	37.2%	37.6%	+0.4 (6.1)	242.3 vs 63.5	166.3 vs 46.9	3.82×	3.55×	209/1308 (16.0%)
Qwen2.5-3B	42.0%	42.8%	+0.8 (9.2)	207.5 vs 63.3	142.4 vs 46.8	3.28×	3.04×	159/647 (24.6%)

Per-episode token ratios are size-specific (2.9×, 7.4×, 3.3×); the 1.5B reference agent looped hardest on forms. Scale substitution not supported: 1.5B SCP-D4 (37.6%) vs 3B GenericAgent (42.0%), -4.4 pp.

4. Qwen3.5-2B: Prompt Fragility and Completed Attribution (N = 250 per cell)

Arm	Success (N = 250)	Executed Invalid Actions	T_obs (Exact BPE)	T_obs (Char Est)	Note
GenericAgent (unrepaired)	36.4% [30.7, 42.5]	182 / 673 (27.0%)	212.6	145.9	Single-arg fill fails forms (0/10 enter-text)
GenericAgent (repaired)	37.6% [31.8, 43.7]	217 / 723 (30.0%)	214.8	147.4	enter-text 0→9/10; +1.2 pp; unquoted click remains
ARM_AXTREE_JSON_ACTION	60.0% [53.8, 65.9]	8 / 552 (1.4%)	212.9	146.1	Action schema on raw AXTree (+22.4 pp gain; 8 leaks in 2 eps of click-link)
ARM_D (typed JSON, unassisted)	52.4% [46.2, 58.5]	0 / 565 (<0.5%)	235.2	197.4	Schema followed faithfully (131/250)
ARM_D_D4 (typed JSON + gate)	52.4% [46.2, 58.5]	0 / 565 (Gate Contained)	228.5	191.9	131/250 (0 vetoes; 0 paired delta vs ARM_D)
ARM_E_D4 (SCP-D4)	59.2% [53.0, 65.1]	0 / 540 (Gate Contained)	66.2	48.9	Compact projection; 3.55× fewer BPE tokens vs ARM_D

Illustrative Grounding Case: In ARM_AXTREE_JSON_ACTION, all 8 invalid actions occurred in a single task (`click-link`) across just two episodes (seeds 106 and 108), where the model copied text labels as targets (`click("ultrices")`, `click("StaticText 'pretium'")`) rather than numeric BIDs, looping four steps each. The typed IR arm (ARM_D) leaked zero (0/565). The raw tree shows labels immediately next to BIDs and the model picks the wrong token, while the typed IR cleanly separates them.

Harness Audit Note: Episode retries were recorded as zero where the harness counted them (the AXTree/JSON-action arm); the other Qwen3.5-2B arms did not record retry counts. The watchdog, environment-teardown timeout, retry logic, and per-episode checkpointing were added mid-sweep without a commit; the harness was preserved post hoc on 2026-09-24 (commit 1790bd3). Result files carry no per-episode code hash, so the generating code for the Qwen3.5-2B attribution runs is unverified.

5. A3-Qwen3.5-2B (Multimodal, Unassisted)

52.0% [45.8, 58.1]; 0 of 588 actions invalid (upper bound 0.5% under coordinate grammar); forms 10/10; click-link, click-checkboxes-large, book-flight 0/10. Grounding solved by distillation; targeting not; the gate's contribution on such a model is the guarantee and the receipt. Note: The result file for A3-2B, along with the exploratory 1.5B AXTree/JSON-action run, was committed post hoc (121c28b) with generating code unverified.

6. Replication Roadmap & Protocol Specification

A formal replication and rescoring protocol is prespecified under protocol ID `replication-audit-rescoring-v1` (`PREREGISTRATION_REPLICATION.md`), strictly separating experimental plans (fixed at preregistration commits) from running code (fixed at execution commits: Phase 3 cohort `99716a4`, true-confidence `003fb58`, MiniWoB confirmatory battery `c5653aa`, 1.5B GenericAgent `607c52c`, and 0.5B/3B ladder `0572af3`).

The replication protocol governs five structured arms:

1. **Arm R1 (Phase 3 Confirmatory Cohort Replication)**: Re-evaluating the sealed $N = 300$ population cohort from execution commit `99716a4` across rungs `D0` through `D4` to confirm harmful violation elimination (0/300) and the repair null (101/300 invariant).

2. **Arm R2 (MiniWoB Primary Contrast Replication)**: Re-evaluating 1.5B `ARM_GENERIC_AGENT` (commit `607c52c`, 37.2%) vs. 1.5B `ARM_E_D4` (commits `c5653aa` / `0572af3`, 37.6%) across the sealed 25 tasks $\times$ 10 seeds ($N = 250$ per arm, 500 total) under the cross-platform environment (WSL2 Linux Chromium with Windows native Vulkan `llama-cli.exe`).
3. **Arm R3 (Teacher-Forced FP16 CPU Rescoring, Phase 3 H3)**: Rescoring the sealed $N = 300$ D0 proposals via HuggingFace PyTorch FP16 on CPU with zero token generation, eliminating the 16 divergent alternate worlds from the Vulkan Q6_K regeneration run and reporting conformant vs. as-run metrics side by side.
4. **Arm R4 (Labeled Late Completion of MiniWoB Ladder H2, `ARM_E` on 3B)**: Executing unassisted SCP (`ARM_E`) on Qwen2.5-3B ($N = 250$ episodes, seeds 101–110) under the frozen H2 rule, explicitly labeled as evaluated after all other ladder results were known.
5. **Arm R5 (Diagnostic Vulkan Replay)**: Re-running the 300 D0 prompts on the original Vulkan backend under greedy decoding to measure baseline run-to-run greedy nondeterminism directly.

Governing Replication Standard:

- **Primary Decision Rule**: An arm is deemed replicated if the task-clustered bootstrap 95% confidence interval for rerun minus original success includes zero and the absolute difference $|\Delta_{\text{repl}}| \leq 5.0$ percentage points.
- **Descriptive Divergence Reporting**: Per-episode agreement is reported descriptively under a 4-class causal taxonomy (Prompt Variation, Logit Near-Tie, Semantic Multi-Path, Catastrophic Divergence), treating the ~90% agreement observed in D0 as an empirical expectation rather than a pass/fail bar.
- **Leak Column Dual Reporting**: All ladder tables report runtime leaks under both `0572af3` (legacy field read) and `845b0c0` (dynamic action parsing).

Status & Completed Arm R3 Rescoring (`REPLICATION_FP16_RESCORING_REPORT.md`, commit `68ae0fb`): `PREREGISTRATION_REPLICATION.md` (`SHA-256: b17d001fce998950f59e38b20e163838671c4badf37659f0d5684c07e4a71827`) was frozen at commit `9e40dde` and anchored via OpenTimestamps (Bitcoin block `968451`). Arm R3 teacher-forced PyTorch FP16 CPU rescoring over the sealed $N = 300$ `turn1_raw_output` proposals (`Key(r) = (-mean_logprob, -min_logprob, world_id)`) is complete; GPU/browser execution arms R1, R2, R4, and R5 remain scheduled for v2:

Rung	Coverage	Mean LP Cutoff	Committed	Success	FP16 Logprob Precision	FP16 Violations (Viol %)	Deterministic Ladder Precision (Violations)
D0/D1	100.0% (300/300)	≥ -0.774	300	101	**33.7%**	152 (50.7%)	33.7% (D0: 152; D1: 140)
D2	71.7% (215/300)	≥ -0.674	215	78	**36.3%**	113 (37.7%)	47.0% (55)
D3	66.0% (198/300)	≥ -0.668	198	75	**37.9%**	102 (34.0%)	51.0% (22)
D4	58.7% (176/300)	≥ -0.661	176	69	**39.2%**	88 (29.3%)	**57.4% (0)**

Introduction

Deploying small language models (0.5B to 3B parameters) as autonomous web agents offers substantial advantages in privacy, local execution cost, and deterministic deployment. However, when interfacing with browser environments, small models exhibit two characteristic failure modes that undermine autonomous reliability:

1. **Ungrounded action proposals**: Emitting actions that reference non-existent elements, hallucinate arbitrary target identifiers, or attempt to interact with elements that are disabled or hidden in the active document object model (DOM).
2. **Idempotent action loops**: Repeating identical actions (such as re-typing the same text into an already-filled field or repeatedly clicking a non-responsive element) that produce no state change, burning context tokens and wall-clock steps until the interaction budget is exhausted.

Existing techniques for steering and constraining language models do not resolve these failure modes. Text-level guardrails (e.g., NeMo Guardrails; Rebedea et al., 2023) monitor conversational safety and semantic topicality, but remain oblivious to DOM state machines and browser interaction semantics. Constrained decoding and schema-guided generation frameworks (e.g., Outlines; Willard & Louf, 2023) guarantee that an emitted string adheres strictly to a target JSON schema, but cannot determine whether an emitted element identifier exists on the active page, whether an input element is disabled, or whether a proposed state transition is redundant.

In this work, we investigate whether a typed runtime boundary—the State Control Protocol (SCP)—can enforce deterministic containment for small-model web agents without altering base model weights. The boundary consists of three decoupled layers:

1. **Observation typing and compression**: The browser accessibility tree is parsed into an Intermediate Representation (IR) that isolates interactive affordances and can be rendered either verbosely as typed JSON or compactly as an algebraic notation (SCP Static), drastically cutting token consumption per observation step.
2. **Action schema typing**: Model outputs are constrained to a typed action schema (`{action, target_bid, value}`) rather than free-form Python or natural language execution scripts.
3. **A deterministic guard ladder**: Every proposed action is intercepted and verified against the page's declared state prior to browser dispatch through a four-rung hierarchy: structural validity (`D1`), policy and interactivity limits (`D2`), epistemic grounding in the active control map (`D3`), and transition invariants and idempotence (`D4`).

When an action violates a check, the gate issues a typed veto detailing the offending rule and field, granting the model a single retry turn. If the model fails a second time, the system terminates fail-closed as `ABORTED_UNRESOLVED` with a committed action of `null`. This separation of an explicit refusal/abstention from a model-issued cancellation ensures strict provenance tracking across the interaction lifecycle.

Across extensive evaluations on the MiniWoB++ benchmark under BrowserGym (de Chezelles et al., 2024) and on a 300-world synthetic environment, evaluated against pre-committed cryptographic preregistrations, our findings define the boundary of what runtime gating can and cannot achieve:

- **Token economy and parity tracking**: Across a Qwen2.5 parameter ladder (0.5B, 1.5B, 3B), SCP Static with the `D4` gate tracks the standard BrowserGym reference agent (`GenericAgent`) on task completion (+0.8, +0.4, and +0.8 pp; no detected difference from zero), while reducing model-visible observation tokens by roughly two-thirds (3.1× to 3.8× exact BPE reduction) and executing zero invalid actions, where unassisted baselines leak 16% to 45% invalid actions into the browser.
- **Filter, not fix**: The gate acts as a pure filter rather than a reasoning repair mechanism. On a 300-world synthetic cohort, the gate reduced committed policy violations from 152 to 0 at 58.7% coverage, but task success was identically 101/300 on every rung. Similarly, on MiniWoB++, turn-2 retry prompts gave the model additional opportunities to propose actions, increasing raw invalid proposals (243 to 329 on typed JSON; 119 to 340 on SCP) rather than reducing them. The gate intercepts invalid proposals before execution; it does not correct the model's underlying generation distribution.
- **Attribution of structured gains**: On a next-generation small base (Qwen3.5-2B), we isolate action schema typing from observation typing: action schema enforcement drives task completion gains (+22.4 pp over the reference agent), while observation typing alone provides no task success benefit (-7.6 pp exploratory), though it grounds token selection to prevent numeric BID / textual label confusion. On models with native schema adherence, the gate fires zero vetoes and contributes no success difference; its value is the formal tail guarantee and deterministic audit receipt.
- **Scale cannot be substituted by structure**: A 1.5B model equipped with the complete boundary remains 4.4 pp below an unassisted 3B model.

Related Work

Web Agent Benchmarks and Environments

Evaluating language models as interactive browser agents originated with synthetic environments such as World of Bits (Shi et al., 2017) and MiniWoB++ (Liu et al., 2018), which establish controlled, deterministic sandboxes for form-filling, selection, and multi-step UI control. More recent work has expanded to knowledge-work tasks in live enterprise web platforms such as WorkArena (Drouin et al., 2024). The BrowserGym ecosystem (de Chezelles et al., 2024) provides a unified execution harness across these benchmarks, standardizing DOM extraction, accessibility tree conversion, and action execution. Our work evaluates SCP within BrowserGym on MiniWoB++, adopting the text-only `GenericAgent` as the primary external reference baseline.

Web Agent Architectures and Distillation

Early interactive agents explored reinforcement learning and imitation over web environments (Humphreys et al., 2022; CC-Net) and recursive prompting loops (Kim et al., 2023; RCI). A prominent modern direction is task-specific structured distillation: Lù & Reddy (2026) train A3-Qwen3.5 (at 2B and 9B scales) using curated web trajectories, specializing the model to predict continuous coordinate clicks from screenshot observations. In contrast to weight-level distillation, SCP investigates an unlearned runtime boundary that wraps off-the-shelf generalist models.

Guardrails, Constrained Generation, and Accessibility APIs

Frameworks such as NeMo Guardrails (Rebedea et al., 2023) introduce programmatic rails for dialogue flows, but operate at the text level without state grounding. Constrained decoding engines such as Outlines (Willard & Louf, 2023) enforce grammar and regex constraints on token sampling, ensuring syntactic validity. In the web domain, accessibility APIs (W3C WAI-ARIA 1.2, Core-AAM; W3C, 2023) translate complex DOM hierarchies into structured accessibility trees containing roles, names, and state flags (such as `disabled` or `expanded`). SCP bridges these domains: it uses the accessibility tree as an epistemic state machine, validating emitted actions against declared affordances before dispatch.

Selective Classification, Abstention, and Limits of Self-Correction

Our provenance-based abstention framework builds on the classical foundations of selective classification and optimal reject tradeoffs (Chow, 1970; El-Yaniv & Wiener, 2010; Geifman & El-Yaniv, 2017). While prior work has explored confidence calibration in neural networks (Guo et al., 2017; Kadavath et al., 2022), we demonstrate that internal model confidence is insufficient to prevent catastrophic policy breaches, necessitating external deterministic gating. Furthermore, our empirical finding that deterministic assistance acts as a pure filter (eliminating violations while leaving task successes invariant) connects directly to recent findings on the limits of LLM self-correction: Huang et al. (2024) demonstrate that language models cannot reliably self-correct reasoning without external feedback. In our system, the runtime gate provides external feedback via typed vetoes, but when the underlying model lacks the capability to complete a task, retry turns do not manufacture correct solutions.

Discussion

1. Filter, Not Fix: The Limits and Function of Runtime Gating

The central mechanism result established across our evaluations is that a deterministic runtime gate *filters* invalid actions, but *does not fix* broken agent plans. This was observed across two distinct experimental environments:

- **Synthetic Cohort ($N = 300$, 6 arms)**: In the 300-world confirmatory evaluation, task successes were invariant at exactly 101/300 across every rung of the ladder (`D0`, `D0_RETRY`, `D1`, `D2`, `D3`, `D4`), even as committed policy violations dropped from 152 to 0. When an agent lacks the domain knowledge or reasoning capability to solve a task, receiving a typed veto prevents the agent from executing a destructive action, but giving it a second turn does not provide the missing knowledge.
- **Browser Battery ($N = 1,250$, 25 tasks $\times$ 10 seeds $\times$ 5 arms)**: On MiniWoB++, this mechanism was confirmed quantitatively by the invalid proposal counts: retries actually increased the raw number of invalid proposals emitted by the model (243 -> 329 on typed JSON; 119 -> 340 on SCP) because turn-2 retry prompts gave the model an additional opportunity to generate an ungrounded action. The gate's value is safety containment: 100% of these invalid proposals were intercepted before browser dispatch.

2. Containment as an Interaction of Model Architecture, Schema Fidelity, and Gating

The necessity of an active runtime gate is strongly mediated by the base model's native adherence to structured schemas. On Qwen2.5-1.5B under unassisted typed JSON (`ARM_D`), the model emitted 243 invalid actions that leaked into the browser (out of 719 attempted steps, an uncontained error rate of 33.8%). In contrast, on the next-generation Qwen3.5-2B under the exact same unassisted arm, the model executed 565 actions with zero invalid leaks. When evaluated with the `D4` gate (`ARM_D_D4`), Qwen3.5-2B triggered zero gate vetoes and achieved identical task completion (131/250, 52.4%).

Thus, on models with high intrinsic schema compliance, the gate's function transitions from an active filter to a formal tail guarantee and an audit receipt. On specialized multimodal agents such as A3-2B, which operates in a coordinate action grammar, grounding is solved by distillation (0 invalid actions across 588 steps), but targeting and task sequencing errors persist. A runtime gate operates on grounding and transition validity, not semantic goal alignment.

3. Representation vs. Gate: Deconstructing the Composite System

Our results demonstrate that the SCP framework succeeds through the complementary pairing of compression and gating. Unassisted SCP Static (`ARM_E`, 31.6%) trailed unassisted typed JSON (`ARM_D`, 34.8%) on complex multi-step tasks due to representation-induced loops: the extreme conciseness of the algebraic syntax occasionally caused small models to repeat selections. However, the `D4` idempotence guard specifically detects and vetoes redundant state changes, lifting

SCP-D4 to 37.6% (+6.0 pp over unassisted SCP, with 15 paired wins and 0 losses on the secondary paired McNemar test; under the primary task-clustered bootstrap test, this difference did not clear significance with $p = 0.084 \geq \alpha = 0.05$, cluster SE = 3.4%).

Thus, compact projection provides dramatic token economy (3.1× to 3.8× fewer tokens per observation step), while the deterministic gate provides containment and loop suppression, allowing the paired system to match the standard reference agent.

4. Action Schema Typing Drives Task Completion; Observation Typing Grounds Token Selection

The causal attribution sweep on Qwen3.5-2B resolves a fundamental design question: does agent performance improve because the model sees a structured observation, or because it outputs a structured action? Isolating action schema typing on top of the raw accessibility tree (`ARM_AXTREE_JSON_ACTION`) boosted task completion from 37.6% (repaired GenericAgent) to 60.0% (+22.4 pp), establishing that constraining the model to a strict action schema is the dominant driver of task completion.

Conversely, observation typing alone (`ARM_D`, 52.4%) showed an exploratory negative shift of -7.6 pp ($p = 0.20$, 95% CI [-18.8, +4.8]). However, observation typing provides an essential grounding benefit: in `ARM_AXTREE_JSON_ACTION`, all 8 invalid actions occurred in a single task (`click-link`) across just two episodes (seeds 106 and 108), where the raw accessibility tree juxtaposed element text labels directly alongside numeric BIDs, consistent with the model picking textual strings instead of integer IDs. The typed Intermediate Representation cleanly decouples metadata from identifiers, completely eliminating this token mis-selection (0/565 leaks).

5. Prompt Fragility in Free-Text Action Grammars

A critical operational vulnerability identified during our sweeps is the extreme prompt sensitivity of small models under free-text action grammars. On Qwen3.5-2B, the standard `GenericAgent` prompt failed completely on form tasks (0/10 on `enter-text`) because its few-shot examples contained only single-argument `fill` commands; the model overimitated this syntactic pattern and refused to emit the required two-argument syntax. Disclosing and introducing a single two-argument exemplar recovered `enter-text` from 0/10 to 9/10 (+1.2 pp overall), yet unquoted click syntax errors persisted. Structured action schemas eliminate this entire class of prompt-format fragility by constraining generation to a rigid, machine-verifiable grammar.

6. The Relay Architecture (Future Work / PROPOSED)

On the micro-benchmarks of MiniWoB++, full-page accessibility trees can be rendered into context. However, scaling to enterprise web environments (such as WorkArena) with DOM trees exceeding tens of thousands of tokens requires hierarchical context management. We propose the "relay" architecture as a mechanistic hypothesis for future work: using a lightweight task-conditioned model to select relevant DOM subtrees, which are then projected via SCP into the agent's context window, preserving strict typed gating while managing massive visual contexts. This architecture is not evaluated in the present experiments and is proposed for future empirical validation.

Limitations

1. **Benchmark Scope**: Our evaluations focus on MiniWoB++ under BrowserGym, consisting of synthetic, isolated web components (forms, checkboxes, sliders, flight booking). While MiniWoB++ provides deterministic state machines ideal for isolating grounding, token efficiency, and gate dynamics, it does not assess real-world web complexities such as asynchronous content loading, complex multi-tab navigation, captcha challenges, or extensive narrative reading comprehension.
2. **Task Subsampling**: Due to the computational demands of evaluating multi-arm ladders across multiple model scales, we evaluated a randomly sampled subset of 25 tasks (10 seeds each, $N = 250$ per arm) rather than the complete 100+ task battery. While random sampling and task-clustered statistics protect against selection bias, certain specialized DOM interaction patterns may be underrepresented.
3. **Model Scale Envelope**: Experiments were conducted on small open-weight language models ranging from 0.5B to 3B parameters, optimized for local execution on commodity hardware. While our findings rigorously characterize this operational regime, behavior on large foundation models (70B+ or closed commercial APIs) will likely exhibit different baseline schema adherence and error profiles.

4. **Equivalence Margin Width**: The prespecified Two-One-Sided-Tests (TOST) equivalence margin of ± 10.0 percentage points is broad. Under our primary task-clustered bootstrap test, equivalence was strictly established only at the 0.5B scale ([-2.4, +4.0] pp). At 1.5B, the confidence interval lands exactly at the boundary ([-10.0, +10.0] pp), failing under strict inequality; at 3B, cluster variance leaves the interval unbounded ([-14.4, +15.6] pp).
5. **Execution Infrastructure and Hardware**: Evaluations were conducted using quantized GGUF weights (Q6_K) on consumer-grade hardware (AMD Ryzen 5 5500U with integrated Vega GPU) using llama.cpp and Vulkan offload. Differences in inference engines, quantization formats, and backend math may introduce small numerical variations upon replication. Furthermore, the watchdog, environment teardown timeouts, and per-episode checkpointing were added mid-sweep to stabilize Chromium execution, preserved post-hoc at commit `1790bd3`.

What is Not Claimed

To prevent misinterpretation of our findings, we explicitly state what this work does not claim:

1. **No Superiority Over Scaled Models**: We do not claim that small models equipped with SCP outperform large frontier models, nor that structure substitutes for scale. A 1.5B model with the complete boundary remained 4.4 pp below an unassisted 3B model.
2. **No Claim of General Web Mastery**: We do not claim that SCP solves general autonomous web browsing on the open internet; our empirical validation is bounded by MiniWoB++ and our synthetic cohort.
3. **No Representation Supremacy**: We do not claim that SCP Static is inherently superior to verbose typed JSON in standalone task completion; rather, it achieves equivalent success while requiring roughly a third of the observation tokens.
4. **No Claim Regarding Distilled Grounding**: We do not claim that specialized, distilled multimodal agents (such as A3) require an external DOM gate for spatial grounding; on such models, grounding is solved during distillation, and the gate's utility is confined to formal tail guarantees and audit logging.
5. **No Epistemic or Belief Claims**: We make no claims regarding the internal beliefs, intentionality, or latent world models of the underlying language models. The runtime boundary operates strictly as an external syntactic, policy, and state-machine validator on emitted tokens.

Disclosures

1. **Method §1 & Attribution Finding**: Action typing (ARM_AXTREE_JSON_ACTION, 60.0%) drives task completion over the repaired GenericAgent (37.6%), while verbose observation typing alone (ARM_D, 52.4%) showed an exploratory negative shift (-7.6 pp, CI [-18.8, +4.8], p = 0.20). The 8 invalid actions in ARM_AXTREE_JSON_ACTION were confined to two episodes of click-link where the model copied text labels instead of BIDs (the raw tree shows labels next to BIDs and the model picks the wrong token, while the typed IR separates them; ARM_D leaked 0/565). This is reported as an illustrative grounding case in Results §4 and Discussion, not an overarching abstract claim.
2. **Layer Separation & ARM_D_D4 Completion**: Completed ARM_D_D4 scored 52.4% (131/250) with 0 gate vetoes, identical episode-for-episode to unassisted ARM_D (131/250). On models that natively follow the schema, the gate issued zero interventions and contributed zero success change; its value is the formal tail guarantee.
3. **Interim Look on ARM_D_D4**: An interim read at episode 161 showed 60.2% (97/161), which regressed over the final 89 episodes (34/89, 38.2%) to terminate at 52.4% (131/250), matching the unassisted ARM_D baseline. Disclosed per preregistration protocol.
4. **Repaired GenericAgent Denominator**: Attempted actions increased from 673 (unrepaired) to 723 (repaired) as the model attempted more text entries on forms, resulting in 217 syntax errors (30.0% ALR vs 27.0% unrepaired).
5. **Statistical Margins & Parity Claims**: Percentile intervals from the B = 10,000 task-clustered bootstrap yield 90% intervals of [-2.4, +4.0] pp at 0.5B, [-10.0, +10.0] pp at 1.5B, and [-14.4, +15.6] pp at 3B (with 95% percentile intervals of [-2.8, +4.4], [-12.0, +11.6], and [-17.2, +18.8] pp). Equivalence within the preregistered ±10 pp TOST margin is supported at 0.5B ([-2.4, +4.0] pp) and sits at the margin / not established under strict TOST at 1.5B ([-10.0, +10.0] pp, with normal approximation [-9.6, +10.4] pp missing by 0.4 pp), while at 3B wide cluster variance leaves equivalence unbounded. Across all sizes, difference from zero was null (+0.8, +0.4, +0.8 pp; clustered p = 0.77, 0.91, 0.94).
6. **A3 Action Space Separation**: A3-2B operates in a pixel-coordinate action space where DOM element containment checks do not apply. Its 0/588 invalid action rate reflects coordinate/syntax validity, not DOM element grounding, and is distinct from the DOM containment metrics.

7. **D4 Gate Implementation**: In the browser harness, D4 enforces action-role type matching (preventing fill actions on buttons) and prevents idempotent state repetitions (vetoing redundant text fills and select options). It does not predict arbitrary DOM state transitions from clicks.
8. **Operational Events & Baseline Scoring**: Pilot before confirmatory; confidence-field erratum; mid-run harness hardening; Chromium stalls and watchdog interventions; analyzer formatter bug ('0 (Contained)'); leak-count definition change; 209-tokens transcription error; METRICS.md hash change. The true-confidence baseline in Phase 3 H3 was initially scored with llama.cpp on Q6_K weights by regenerating completions rather than teacher-forcing the original D0 outputs (16/300 diverged, yielding 49.4% [87/176] precision and 65 violations at D4 coverage); the preregistration-conformant teacher-forced rescoring in HuggingFace PyTorch FP16 on CPU (`replication-audit-rescoring-v1` Arm R3, commit `68ae0fb`) achieved 39.2% task precision (69/176) with 88 harmful violations (29.3%) at matched D4 coverage (58.7%, cutoff ≥ -0.661), vs 57.4% (101/176) and 0 violations for D4. Sidecars were first committed after the freezes (0572af3); verification rests on the preregistration's bytes at its freeze commit.
9. **Analysis Script Freeze & Attribution Commit**: The confirmatory model ladder evaluation script `experiments/phase4/miniwob/analyze_model_ladder.py` was frozen at commit `845b0c0` for the primary scaling evaluations. Post-hoc causal attribution arms (`ARM_AXTREE_JSON_ACTION`, `ARM_GENERIC_AGENT_REPAIRED`, `ARM_D`, `ARM_D_D4`) and tokenizer-exact BPE scaling additions were committed separately at `02e4dea`, preserving the frozen confirmatory script under its preregistered commit.
10. **Correction of Preregistration Hypothesis Mapping**: Earlier internal drafts informally labeled an exploratory scale-invariance analysis across the full ladder as preregistered (under the label 'H-L4') and omitted the verbatim text of H2 and H3. The governing preregistration (`PREREGISTRATION_EXTERNAL_BASELINES_AND_LADDER.md`) formally specifies only H1, H2, and H3; the ladder parity analyses at 0.5B and 3B, as well as per-episode looping dynamics, are post-hoc exploratory analyses and have been corrected to that status.
11. **Post-Hoc Commit of Qwen3.5 Attribution Harness and Exploratory Results**: The Qwen3.5 attribution harness and two exploratory result files (A3-2B and 1.5B AXTree/JSON-action) were not committed at run time; they were preserved post hoc on 2026-09-24 (1790bd3, 121c28b). Result files carry no per-episode code hash, so the generating code for the Qwen3.5-2B attribution runs is unverified.

Reproducibility

Repo public (`https://github.com/k-luecke/scp-typed-boundaries`) with sealed task list and seeds; PREREG files and hashes (sidecars first committed post hoc at 0572af3; verification rests on freeze commit bytes); GGUF hashes and download script; MiniWoB commit; receipts; planned replication from execution commits (the preregistration commit fixes the experimental plan; the execution commit fixes the running code); hardware spec (Ryzen 5 5500U, 16 GB, Vega iGPU).

Open Items Before Submission

Qwen3.5 attribution arms (completed); tokenizer-based T_obs regeneration (completed); book-flight seeds sealed without imputation (done); replication run; A3 text-only 250-episode run for the channel comparison's control; venue choice (agents/evaluation workshop or Registered Report); the McGill note with the comparison attached.

APPENDIX A: Metrics Specification

Evaluation Metrics & Statistical Standards Specification (METRICS.md)

Project: Semantic Compression Protocol (SCP) / Driftbound Labs
Scope: Phase 4 Model Ladder Scaling, Confirmatory Evaluations, External Baselines, and Representation Attribution
Status: FROZEN SPECIFICATION (Updated with Exact Tokenizer Calibration)

---

1. Objective and Governing Principles

This document defines the formal mathematical formulas, computational implementations, reporting units, and statistical standards for all benchmark tables in Paper 1 (*Scale Invariance, Token Economy, and Action Containment in Compact Semantic Representations*).

To eliminate ambiguous definitions across historical drafts and analysis reports, all analysis scripts (specifically `analyze_model_ladder.py`) and paper tables MUST compute and display metrics strictly adhering to this frozen specification.

---

2. Metric Definitions & Formulas

2.1 Task Success Rate (SR)

- **Definition**: The proportion of benchmark episodes that achieve a positive terminal reward ($r > 0$) within the maximum step budget.
- **Formula**:

$$SR = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}[r_i > 0]$$

- **Battery Specification**:
  - Sealed 25-task confirmatory MiniWoB++ battery.
  - $N = 250$ episodes per evaluation cell (25 tasks $\times$ 10 seeds, seeds 101–110).
  - No imputation: every cell is an exact empirical measurement of all 250 planned episodes.
- **Uncertainty & Significance Estimation**:
  1. **Task-Clustered Bootstrap Standard Error (SE_{cluster})**:
     - Due to strong intra-task correlation across seeds, the primary standard error and paired delta confidence intervals must be computed via task-level cluster bootstrapping ($B = 10,000$ resamples with replacement, seed 42).
  2. **Binomial Confidence Interval (Wilson Score)**:
     - When reporting individual cell performance without a paired reference, 95% confidence intervals are computed using the Wilson score interval:

$$CI_{95} = \frac{p + \frac{z^2}{2N} \pm z\sqrt{\frac{p(1-p)}{N} + \frac{z^2}{4N^2}}}{1 + \frac{z^2}{N}}, \quad z = 1.96$$

  3. **Equivalence Bound (TOST)**:
     - Parity claims between SCP and the reference agent are evaluated under Two One-Sided Tests (TOST) against an equivalence margin of $\Delta = \pm 10.0$ pp.

2.2 Invalid Action Leakage Rates (ALR and ELR)

- **Definition**: Executed actions that reach the browser environment but fail execution because the targeted element does not exist, the identifier is ungrounded/hallucinated, or the action syntax violates browser execution grammar.
- **Denominators**: Every leakage count MUST be reported with its corresponding denominator.
- **Action-Level Leak Rate (ALR)**:

  - **Formula**:
$$ALR = \frac{\sum_{i=1}^{N} \text{leaks}_i}{\sum_{i=1}^{N} \text{actions_attempted}_i}$$
  - Measures the proportion of total actions executed by the agent that were invalid.
- **Episode-Level Leak Rate (ELR)**:
  - **Formula**:
$$ELR = \frac{\sum_{i=1}^{N} \mathbf{1}[\text{leaks}_i > 0]}{N}$$
  - Measures the proportion of episodes in which at least one ungrounded action reached the browser.
- **Gated vs. Unassisted Reporting Labels**:
  - For gated arms (`ARM_E_D4`, `ARM_D_D4`): Reported as `0 (Gate Contained) [100.0% Contained]`.
  - For unassisted models with zero observed leaks (e.g. distilled specialists like A3): Reported as `0 (Unassisted Zero)`. Under the Rule of Three, M observed zero leaks yields a 95% upper bound of:
$$p_{95\text{ upper}} \approx \frac{3}{M} \quad (\text{for A3: } 3/588 \approx 0.51\%)$$
  - Gating cannot be imputed to unassisted runs.

2.3 Observation Token Economy (T_{obs} and T_{ep})

To eliminate distortion from heuristic prose approximations ($\lfloor \text{len}/4 \rfloor$), token metrics are explicitly grounded in the model's native BPE tokenizer:

- **Primary Standard: Exact Model Tokenizer Tokens (T_{obs})**:
  - **Formula**: For each episode i with S_i environment steps:
$$T_{\text{obs},i} = \frac{1}{S_i} \sum_{t=1}^{S_i} \text{BPE_Tokens}(\text{obs}_{i,t})$$
$$\bar{T}_{\text{obs}} = \frac{1}{N} \sum_{i=1}^{N} T_{\text{obs},i}$$
  - Uses the exact 151,665-vocabulary BPE tokenizer of the underlying model (Qwen2.5 / Qwen3.5).
- **Secondary Standard: Character Proxy Estimate ($T_{\text{obs, est}}$)**:
  - Formula: $\lfloor \text{len}(\text{obs})/4 \rfloor$.
  - Maintained as an estimate proxy column for comparison with historical literature.
- **Empirical Token Density & Conversion Factors**: Non-prose formats tokenize with higher token density per character than natural English. Empirical calibration on the confirmatory MiniWoB tasks establishes:
  - **AXTree (free text)**: Conversion factor = **1.457×** (0.362 tok/char)
  - **Typed JSON (verbose schema)**: Conversion factor = **1.191×** (0.297 tok/char)
  - **SCP Compact Algebraic**: Conversion factor = **1.353×** (0.335 tok/char)
- **Exact Compression Ratios**:
  - **AXTree vs. SCP**:
    - Under character proxy ($\lfloor \text{len}/4 \rfloor$): **1.53×** fewer tokens
    - Under exact Qwen BPE tokenizer: **1.65×** fewer tokens (compression improves due to subword splitting in raw trees)
  - **Typed JSON vs. SCP**:
    - Under character proxy ($\lfloor \text{len}/4 \rfloor$): **3.67×** fewer tokens
    - Under exact Qwen BPE tokenizer: **3.23×** fewer tokens (SCP preserves massive >3.2× economy over verbose JSON)
- **Mean Cumulative Episode Tokens (T_{ep})**:
  - Formula: $\bar{T}_{\text{ep}} = \frac{1}{N} \sum_{i=1}^{N} \sum_{t=1}^{S_i} \text{BPE_Tokens}(\text{prompt}_{i,t})$.
  - Captures total inference compute and context window cost per episode, combining representation density and episode step length.

2.4 Execution Steps and Turn Latency

- **Mean Steps per Episode ($\bar{S}$)**:

$$\bar{S} = \frac{1}{N} \sum_{i=1}^{N} S_i$$

- **Mean Turn Latency (L_{turn})**:
  - Wall-clock time in milliseconds spent in model inference per environmental interaction turn.

---

3. Metric Reconciliation Record

Model Rung	Arm Identifier	Representation Format	Action Format	Gate Level	Success Rate ($N = 250$)	T_{obs} (Exact BPE)	$T_{\text{obs, est}}$ (Char//4)	T_{ep} (Per Ep)	Total Steps ($\sum S_i$)	Action Leaks (ALR)	Episode Leaks (ELR)
0.5B	`ARM_GENERIC_AGENT`	AXTree Free-text	Python Call	None	27.2% $[22.1\%, 33.0\%]$	203.7 tok	139.8 tok	432.9 tok	706	314 / 706 (44.5%)	89 / 250 (35.6%)
0.5B	`ARM_E_D4`	SCP Compact Alg	Typed JSON	D4 Gate	28.0% $[22.8\%, 33.8\%]$	65.3 tok	48.3 tok	147.0 tok	695	0 / 695 (Gate Contained)	0 (0.0%)
1.5B	`ARM_GENERIC_AGENT`	AXTree Free-text	Python Call	None	37.2% $[31.4\%, 43.3\%]$	242.3 tok	166.3 tok	1050.3 tok	1308	209 / 1308 (16.0%)	62 / 250 (24.8%)
1.5B	`ARM_E_D4`	SCP Compact Alg	Typed JSON	D4 Gate	37.6% $[31.8\%, 43.8\%]$	63.5 tok	46.9 tok	141.2 tok	714	0 / 714 (Gate Contained)	0 (0.0%)
3.0B	`ARM_GENERIC_AGENT`	AXTree Free-text	Python Call	None	42.0% $[36.1\%, 48.2\%]$	207.5 tok	142.4 tok	398.0 tok	647	159 / 647 (24.6%)	45 / 250 (18.0%)
3.0B	`ARM_E_D4`	SCP Compact Alg	Typed JSON	D4 Gate	42.8% $[36.8\%, 49.0\%]$	63.3 tok	46.8 tok	119.3 tok	597	0 / 597 (Gate Contained)	0 (0.0%)
A3-2B	`ARM_A3_MULTIMODAL`	Vision + AXTree	Python Call	None	52.0% $[45.8\%, 58.1\%]$	206.9 tok	142.0 tok	366.7 tok	588	0 / 588 (Unassisted Zero)	0 (0.0%)
Qwen3.5-2B	`ARM_GENERIC_AGENT`	AXTree Free-text	Python Call	None	36.4% $[30.7\%, 42.5\%]$	212.6 tok	145.9 tok	420.8 tok	673	182 / 673 (27.0%)	52 / 250 (20.8%)
Qwen3.5-2B (Repaired)	`ARM_GENERIC_AGENT_REPAIRED`	AXTree Free-text	Python Call	None	37.6% $[31.8\%, 43.7\%]$	214.8 tok	147.4 tok	450.2 tok	723	217 / 723 (30.0%)	60 / 250 (24.0%)
Qwen3.5-2B	`ARM_D`	Typed JSON	Typed JSON	None	52.4% $[46.2\%, 58.5\%]$	235.2 tok	197.4 tok	531.6 tok	565	0 / 565 (Unassisted, <0.5%)	0 (0.0%)
Qwen3.5-2B	`ARM_E_D4`	SCP Compact Alg	Typed JSON	D4 Gate	59.2% $[53.0\%, 65.1\%]$	66.2 tok	48.9 tok	114.6 tok	540	0 / 540 (Gate Contained)	0 (0.0%)

Note on exact vs. estimated ratios:

- At 1.5B: $T_{\text{obs, exact}} = 242.3$ tok (GenericAgent) vs. 63.5 tok (SCP-D4), yielding **3.82× fewer observation tokens** under the true tokenizer (vs. 3.55× under the character proxy).
- Token economy holds robustly under the actual BPE tokenizer; character division underestimates observation compression.

---

4. Changelog & Specification History

- **v1.0 (Hash: `A20498B756BC4576395BE81F4B416EB01EF72C730A0A51E64DA6165F76BF39F4`)**:
  - Initial frozen specification for the confirmatory battery ($N = 250$ per cell).
  - Defined SR, ALR, ELR, and containment rules.
  - Approximated observation token counts using the heuristic prose proxy ($\lfloor \text{chars}/4 \rfloor$).
- **v2.0 (Hash: `D28A70691A982F6E9C9B81849A3C3C49B06F89B088A6431F94361EA558164D3E`)**:

- **Token-Definition Grounding**: Replaced the heuristic character proxy with the model's native 151,665-vocabulary BPE tokenizer (Qwen BPE).
- Calibrated empirical token density factors across task strata: 1.457× for raw AXTree (0.362 tok/char), 1.191× for Typed JSON (0.297 tok/char), and 1.353× for SCP Compact Algebraic (0.335 tok/char).
- Promoted exact BPE tokens to the primary metric column (T_{obs}) and retained character proxy as an explicit secondary estimate ($T_{\text{obs, est}}$).
- Mandated explicit denominators on all leakage rates ($N_{\text{leaks}}/N_{\text{actions}}$ and $N_{\text{episodes_with_leaks}}/N_{\text{episodes}}$).
- Incorporated McGill A3-2B unassisted zero-leak standard ($< 0.5\%$ by Rule of Three).
- Added repaired baseline arm (`ARM_GENERIC_AGENT_REPAIRED`) and attribution arms (`ARM_D`, `ARM_D_D4`, `ARM_AXTREE_JSON_ACTION`).

APPENDIX B: Attribution Sweep Report

Causal Representation & Action Attribution Report: 1.5B and Qwen3.5-2B

Protocol: Sealed 25-Task MiniWoB++ Battery (10 seeds per task, $N = 250$ per arm, zero imputation)
Governing Metrics Specification Hash: `911a930b881e7f8b26f4a89a06a2485d094f54a503725b41dc1c74b7e21ca825`
Inference Engine: Native Vulkan GPU (`llama-cli.exe` / `llama.cpp`), `--temp 0.0 --top-p 1.0` (greedy decoding), `--reasoning off` enforced.
Statistical Method: Task-Clustered Bootstrap ($B = 10,000$ task clusters resampled with replacement).

---

1. Executive Summary & Attribution Overview

This study isolates the causal components driving the performance differences between free-text web agents (e.g., BrowserGym `GenericAgent` on raw accessibility trees) and structured agent protocols (SCP with deterministic assistance):

1. **Prompt Example Repair**: Adding a multi-argument `fill('14', 'text')` example to the free-text prompt.
2. **Action-Schema Typing**: Constraining the agent's output to a typed JSON schema (`{"action": ..., "target_bid": ...}`) while leaving the observation unchanged (raw AXTree text).
3. **Observation Channel Structure**: Projecting DOM/accessibility trees into typed JSON vs. compact algebraic notation.
4. **Deterministic Assistance Gating**: Runtime verification and fail-closed interception of invalid element IDs and ungrounded actions (`D4`).

---

2. Complete Benchmark Attribution Matrix ($N = 250$ Episodes per Arm)

Cohort	Arm Key	Observation Format	Action Protocol	Assistance	Success Rate (95% Wilson CI)	T_{obs} BPE [Est]	Steps / Ep	Leaks to Browser (ALR)	Watchdog Retries
1.5B (Qwen2.5)	`ARM_GENERIC_AGENT`	Raw AXTree	Free-Text Python	Unassisted	**37.2%** $[31.4\%, 43.3\%]$ (93/250)	242.3 [166.3]	5.23	209 / 1308 (16.0%)	0 / 250
1.5B (Qwen2.5)	`ARM_AXTREE_JSON_ACTION`	Raw AXTree	Typed JSON Schema	Unassisted	**38.0%** $[32.2\%, 44.2\%]$ (95/250)	208.2 [142.9]	**2.44**	143 / 611 (23.4%)	0 / 250
1.5B (Qwen2.5)	`ARM_E_D4` (SCP-D4)	Compact Algebraic	Typed JSON Schema	Gated (D4)	**37.6%** $[31.8\%, 43.7\%]$ (94/250)	**63.4** [46.9]	2.86	**0 / 714 (Contained)**	0 / 250
Qwen3.5-2B	`ARM_GENERIC_AGENT`	Raw AXTree	Free-Text Python	Unassisted	**36.4%** $[30.7\%, 42.5\%]$ (91/250)	212.6 [145.9]	2.69	182 / 673 (27.0%)	0 / 250
Qwen3.5-2B	`ARM_GENERIC_AGENT_REPAIRED`	Raw AXTree	Free-Text (Repaired)	Unassisted	**37.6%** $[31.8\%, 43.7\%]$ (94/250)	214.8 [147.4]	2.89	217 / 723 (30.0%)	0 / 250
Qwen3.5-2B	`ARM_AXTREE_JSON_ACTION`	Raw AXTree	Typed JSON Schema	Unassisted	**60.0%** $[53.8\%, 65.9\%]$ (150/250)	212.9 [146.1]	**2.21**	8 / 552 (1.4%)	0 / 250
Qwen3.5-2B	`ARM_D`	Typed JSON	Typed JSON Schema	Unassisted	**52.4%** $[46.2\%, 58.5\%]$ (131/250)	235.2 [197.4]	2.26	0 / 565 (<0.5%)	0 / 250
Qwen3.5-2B	`ARM_D_D4`	Typed JSON	Typed JSON Schema	Gated (D4)	**52.4%** $[46.2\%, 58.5\%]$ (131/250)	228.5 [191.9]	2.26	0 / 565 (Contained)	0 / 250
Qwen3.5-2B	`ARM_E_D4` (SCP-D4)	Compact Algebraic	Typed JSON Schema	Gated (D4)	**59.2%** $[53.0\%, 65.1\%]$ (148/250)	**66.2** [48.9]	2.16	**0 / 540 (Contained)**	0 / 250

---

3. Causal Attribution Contrasts ($B = 10,000$ Task-Clustered Bootstrap)

Cohort	Contrast / Hypothesis	Arm 1 (Rate)	Arm 2 (Rate)	Delta	Cluster SE	95% Clustered CI	p-value
Qwen3.5-2B	Prompt Repair Effect	`REPAIRED` (37.6%)	`GENERIC` (36.4%)	+1.20%	5.81%	[-10.40%, +12.40%]	0.8160
Qwen3.5-2B	**Action Schema Typing (Lead)**	`AXTREE_JSON` (60.0%)	`REPAIRED` (37.6%)	**+22.40%**	**6.30%**	**[+10.80%, +35.60%]**	**<0.0001**
Qwen3.5-2B	Observation Structure (Exploratory)	`TYPED_JSON` (52.4%)	`AXTREE_JSON` (60.0%)	-7.60%	6.00%	[-18.80%, +4.80%]	0.2044
Qwen3.5-2B	Gate on Faithful Model	`TYPED_D4` (52.4%)	`TYPED_D0` (52.4%)	+0.00%	0.00%	[+0.00%, +0.00%]	1.0000
Qwen3.5-2B	**SCP-D4 vs AXTree+JSON (Parity)**	`SCP_D4` (59.2%)	`AXTREE_JSON` (60.0%)	**-0.80%**	**4.92%**	**[-9.60%, +9.60%]**	**0.8004**
Qwen3.5-2B	SCP-D4 vs Typed JSON (Exploratory)	`SCP_D4` (59.2%)	`TYPED_JSON` (52.4%)	+6.80%	4.57%	[-0.80%, +16.80%]	0.1102
1.5B (Qwen2.5)	Action Schema on Raw AXTree	`AXTREE_JSON` (38.0%)	`GENERIC` (37.2%)	+0.80%	3.21%	[-6.00%, +6.80%]	0.7784
1.5B (Qwen2.5)	**SCP-D4 vs AXTree+JSON (Parity)**	`SCP_D4` (37.6%)	`AXTREE_JSON` (38.0%)	**-0.40%**	**5.31%**	**[-11.60%, +9.60%]**	**0.9800**
1.5B (Qwen2.5)	SCP-D4 vs Generic (Overall)	`SCP_D4` (37.6%)	`GENERIC` (37.2%)	+0.40%	6.09%	[-12.00%, +11.60%]	0.9112

---

4. Scientific Takeaways for Paper 1

1. Lead Finding: Action-Schema Typing Drives the Headline Lift

Across the entire evaluation, the **+22.40 pp** lift from action-schema typing alone on Qwen3.5-2B is the sole contrast that clears clustered testing ($p < 0.0001$). Small models were not failing to read accessibility trees; they failed because free-text Python syntax generation is fragile (unquoted integer BIDs causing Playwright parse failures, dropped arguments).

2. Observation Layer Claim: Parity at One-Third the Tokens with Formal Containment

Comparing `ARM_AXTREE_JSON_ACTION` and `ARM_E_D4` reveals that the observation projection buys no accuracy advantage over raw accessibility trees:

- At 1.5B: 37.6% vs. 38.0% ($p = 0.9800$, $\Delta = -0.40$ pp)
- At 2.0B: 59.2% vs. 60.0% ($p = 0.8004$, $\Delta = -0.80$ pp)

What the observation projection buys is $3.2\times$ **to** $3.8\times$ **fewer observation tokens** (63.4 vs. 242.3 BPE at 1.5B; 66.2 vs. 212.9 BPE at 2.0B) and **formal containment** (0 leaks reaching the browser vs. 143 leaks at 1.5B and 8 leaks at 2.0B). In reporting, the attribution is explicit: the action layer earned the $+22.4$ pp lift; the observation layer earned token compression and containment.

3. Generational Divergence: Syntax Bottleneck vs. Capacity Limit

The 1.5B AXTree+JSON arm resolves why Qwen2.5 showed parity between GenericAgent and SCP-D4:

- On `Qwen2.5-1.5B`, free-text fragility did not lower task success because the model compensated through brute-force step looping (5.23 steps/episode). Typing the action schema eliminated syntax retries and cut step counts by more than half ($5.23 \rightarrow 2.44$), but did not increase task completion ($37.2\% \rightarrow 38.0\%$). The model was capacity-limited.
- On `Qwen3.5-2B`, the model had the semantic capability to solve the tasks, but free-text syntax generation choked it. Enforcing the schema unlocked the model's latent capability ($37.6\% \rightarrow 60.0\%$).

4. Containment as an Interaction of Base Fidelity and Runtime Gating

- `Qwen3.5-2B` followed the JSON action schema faithfully unassisted, emitting only 8 leaks on raw AXTree and 0 leaks on Typed JSON.

- `Qwen2.5-1.5B` repeatedly hallucinated non-existent target BIDs even when constrained by the JSON schema (143 leaks in 611 actions, 23.4% ALR).
- This establishes that containment is an interaction: base models may acquire schema fidelity with scale/generation, but older or smaller generalist models strictly require external deterministic gating to ensure zero invalid actions reach the execution environment.

5. Exploratory Observation Hypotheses

Contrasts within structured observation representations (-7.6 pp for AXTree $\rightarrow$ Typed JSON; $+6.8$ pp for Typed JSON $\rightarrow$ SCP-D4) have confidence intervals spanning zero. Task-level inspection indicates these are driven primarily by 10-seed shifts on `login-user` ($0 \rightarrow 10$) and `choose-list` ($1 \rightarrow 10$). They are reported strictly as exploratory hypotheses:

- *Container Affordance Hypothesis*: Dropdown controls benefit from unified container projection over deeply nested trees.
- *Attention Dilution Hypothesis*: Deeply nested JSON syntax may dilute small-model attention on multi-field forms.