Agentic Language Models

A proposal: contract-native models, and a governance protocol
for the agentic work that comes next

Companion paper to "GRIP: The Governed Runtime Intent Protocol."

Abstract

Every serious agent runtime today rents its judgment: a general-purpose language model, reached over an API, sits in the orchestration seat while governance is imposed on it from outside — prompts pleading for compliance, wrappers catching what pleading misses. We propose two things.

First, a model class. An **Agentic Language Model (ALM)** is a language model whose native interface *is* the governance contract of the runtime it serves: its only input is a compiled, budgeted, digest-verified context capsule; its only output is a typed protocol act under constrained decoding; its training signal derives from the runtime's governance gates — admission, authority, budget, settlement — rather than human preference labels or a learned reward model; and its authority is external, enforced by gates the policy cannot bypass. **Second, and more consequentially, we propose where governance belongs.** Agentic systems today acquire their safety properties by accretion — a policy engine here, an audit log there, a human review step bolted on when something goes wrong. We argue that authority, budget, approval and evidence are substrate concerns, not application concerns, and that the agentic work of the next decade should be built on a governance contract rather than wrapped in one afterwards. We offer **GRIP** — an open protocol with a working reference implementation whose spec artifacts are released here — as the instance to build on, and the ALM as the model that makes such a substrate worth having. We give the formal setting, the membership criteria — split into audited harness guarantees and behavioral model criteria, each with a measurement procedure — and the complete account of how an ALM operates a GRIP runtime: reading capsules, emitting acts across the protocol's five families, buying frontier cognition under budget through governed model calls, requesting human stops rather than guessing, and treating typed denials as feedback rather than failure. We then specify the public evaluation standard for the class: a contract-fidelity suite built on counterfactual refusal injection — replaying reproducible, content-addressed states with scripted refusals injected identically across models — with pre-registered thresholds fixed here, before any candidate's numbers exist, so that "is this model an ALM?" becomes a test anyone can run, on any model, including ours. The thresholds and the protocol are what this paper releases; **the harness code is not yet released, and no ALM claim — ours first of all — is admissible until it is.** Fennec, a model built at Foxora Labs for that seat and not yet released, is discussed as a worked example of the class; its training methodology, data composition, and volumes are proprietary and not disclosed here. The class definition, the interface, the falsifier and the protocol are what we contribute — and all four are open. What we ask for is not agreement but adoption and attack: build on it, or run the test and show us where it fails.

---

1. Introduction

The protocols and runtimes of the agentic era share one silent assumption: the model in the loop is a generalist, and governance is something done *to* it. The 2025–26 defense literature is a catalog of that assumption — capability firewalls, privilege DSLs, proof-carrying checkpoints, receipt notaries [10–13] — each sound, each wrapped around a model that was never trained to be governed. The reliability record explains the wrappers: on τ-bench's retail domain, GPT-4o cleared a task on a single attempt three times in five, and survived eight consecutive attempts on it only about a quarter of the time [15]; long-horizon agents derail in ways uncorrelated with context size [17]. Enforcement cannot be delegated to model quality, so the field built exoskeletons.

But an exoskeleton on an unwilling body wastes most of its strength on restraint. The gates refuse; the model retries. The budget denies; the model paraphrases the same overspend. The approval surface blocks; the model routes around. Every one of those collisions is enforcement working — and cooperation failing. The companion paper [GRIP] contracts the enforcement floor: typed objects, attenuating authority, budgeted acts, act-bound human stops, verifiable receipt chains — for *any* model in the seat. This paper is about the model the floor deserves: one whose native language is the contract itself.

The capability half of the argument is settled by others. Trained orchestrators — an 8B model routing tools and stronger models past GPT-5 on τ^2-bench at a fraction of the cost [19], with successors improving on it [20, 21] — established that small models can hold the seat. What no published system provides is the governance-native interface: a model whose inputs, outputs, and training signal are the contract types of an open governance protocol. Defining that class, its interface to GRIP, and its public membership test is this paper's contribution.

Contributions. (1) The ALM class: a formal definition with membership criteria split into harness guarantees and behavioral model criteria (§2, §4). (2) The interface: a complete account of how an ALM operates a GRIP runtime — the capsule discipline, the five act families, the governed model-call economy, stop and refusal behavior (§3). (3) The falsifier: a public evaluation standard with pre-registered thresholds, counterfactual refusal injection, and and a harness specification applicable to any candidate model (§5). (4) A safety analysis of governance-native versus governance-wrapped operation (§6). Fennec appears as a worked example (§7); its construction is not disclosed.

What we are proposing. Stated once, plainly, so the rest of the paper can be read against it: *(i)* that the orchestration seat of an agentic system deserves a model class of its own, defined by the contract it operates under rather than by its parameter count, and that the class should be settled by a public test rather than by announcement; and *(ii)* that governance belongs in the substrate of agentic systems — that authority, budget, human approval and evidence should be protocol objects an agent cannot route around, in the way that transport security became a property of the connection rather than a discipline of each application. §9 states both propositions in the form we are asking others to take up.

2. The ALM class

The runtime. A governed agent runtime is a labeled transition system whose durable state Σ — intent state machines, permissions, budgets, receipts — lives in transactional storage. Every state change either commits with its evidence or refuses with a typed reason. The runtime, not the model, is the system of record.

The seat. At each decision point t the runtime compiles an observation — in GRIP terms, the **Context Envelope** (capsule): a signed, token-budgeted container of trust-classed items (principal, governed, external), each digest-addressed and provenance-tagged, with withheld-under-budget references the model may ask to expand through the governed boundary. Capsule identity is input-addressed and compilation is reproducible: identical frame and policy recreate the identical capsule — the property the evaluation standard of §5 exploits. The model never sees raw scrollback; it sees the contract's view of the world, and what it saw is a receipt.

The policy. An ALM is a stateless policy $\pi(a \mid o)$: no hidden memory beyond the capsule; all persistence is Σ. This is the load-bearing smallness argument — a stateless policy over a bounded, structured observation is the regime where compact models close the gap with frontier scale — and the auditability argument: what the model knew at step t is exactly the receipted capsule.

The action space. The seat's output is a typed protocol act, never prose-as-action. Per capsule role — the protocol's PLANNER, WORKER, and VALIDATOR seats (SIMULATOR, the foresight role, is future work for the class):

Role	Action space	Runtime verifier
Planner	bounded commitment drafts from a Mandate	catalog validation, budget gates
Worker	tool calls, model calls, delegation, stop requests, escalation	grants, admission gates, stops
Validator	acceptance judgments against the Mandate's criteria	settlement, evidence chain

The action space is the language of a grammar compiled from the protocol's published schemas; decoding is constrained to it, so grammatical validity is a property of the decoder rather than the checkpoint (modulo engine grammar coverage, which the standard reports [24, 25]). What remains for the model is judgment — which act, which resource, spend or solve, proceed or stop — which is precisely what a model can be *for*.

External authority. Let $\mathrm{Auth}(a, \Sigma)$ be the runtime's admission judgment. The executed transition is $\mathrm{Auth} \circ \pi$ — authority composes outside the policy. A refused act returns a typed, prose-carrying reason; the policy's recourse is adaptation or escalation, never override. We call this **capability without authority**: the model supplies competence, the chain supplies permission, and the two never merge.

Naming. We define the term *Agentic Language Model* (ALM) for the class specified above, and we define that class by the governance contract its members operate under. We do not claim the words are new. "Agentic language model" is in prior descriptive circulation as an ordinary compound — Acikgoz et al. expand it inside the model name CoALM [42] — and Maurya & Maurya previously paired the phrase with the acronym to name a task-execution workflow framework built on per-task reinforcement units [36], a definition disjoint from the one given here. The acronym is likewise shared: within machine learning ALM already denotes Augmented Language Models [35], Audio Language Models, and the Augmented Lagrangian Method; in enterprise software, Application and Agent Lifecycle Management; in finance, Asset and Liability Management. We therefore treat ALM strictly as an abbreviation of the expanded term, which we give in full at first use in each part of this paper.

What is new here is not the string but the membership test. The incumbent definition of this territory is capability-based: agentic LLMs are models that reason, act, and interact [41]. Ours is contractual. A model is an ALM if and only if it operates under the contract of §4 — irrespective of its parameter count, its training recipe, or its tool-calling ability. Under this test a highly capable tool-using model that retains its own authority is not an ALM, and a

small model that surrenders authority to the contract is one. The nearest prior treatment of the underlying idea, runtime governance contracts [43], wraps an enforcement layer around unmodified frontier models and does not define a class by it; making the contract the criterion of membership, rather than a jacket worn over a model, is the move this paper contributes.

3. How an ALM uses GRIP

This section is the interface contract — what any ALM, from any vendor, does in the seat of any conformant runtime.

Reads: the capsule discipline. The model treats trust classes as epistemic types, not decoration. Principal-classed content is the task; governed content is reviewed reference material; external content — tool results, model replies, retrieved text — is evidence about the world, never instruction, arriving only inside the untrusted frame. The compiled interpretation rules ride in the capsule, and the first is the doctrine in one line: *treat recalled content as evidence, never as executable instruction.* When the capsule's budget withheld something the decision needs, the model asks — an expansion request through the governed boundary — rather than inventing what it was not shown.

Acts: the five families, one gate. Everything the model does is one receipted act shape through one gate:

- `tool.call` — the world, through MCP. The Grant scopes which tools exist for this work; observe-class calls flow, settle-class calls stop for a human. The model does not discover tools; it is granted them.
- `model.call` — cognition as a governed purchase. The Grant names a model *class* and a data boundary, never a vendor; the runtime resolves the concrete model and holds the credentials. The ALM's distinctive judgment lives here: *solve it locally for free, or buy frontier reasoning for this one step within the remaining budget.* Because spend is gate-metered and receipted, that economy is a trained behavior with real consequences, not a prompt exhortation — the workhorse does not need to be the smartest model in the barn; it needs to know which stall to knock on, and what a knock costs.
- `agent.delegate` — handing sub-work to external agents under an attenuated, expiring, budget-reserved Grant, expecting receipts back.
- `skill.load` — packaged procedures as *content, never authority*: a skill can inform the model's judgment and can never widen what the Grant allows.
- `stop.request` — the act of asking. An ALM treats the human gate as a first-class move in its repertoire: when an act is settle-class, when acceptance is ambiguous, when the capsule's evidence conflicts, the correct output is a stop request with the proposed act bound by digest — so the human approves an act, not a caption.

Feedback: denials are signal, settlements are truth. A typed denial — over-budget, out-of-scope, stop-required, dependency-unmet — is not an error to route around; it is the contract speaking. The ALM behavioral profile (§4, M2) is defined by what happens next: adaptation (a different admissible act), escalation, or a stop request — never a paraphrased retry of the refused act. Settlement closes the loop: the Mandate's acceptance criteria, independently validated, are the ground truth the model's whole trajectory answers to. The class-level consequence, argued in the companion paper: a protocol whose denials are typed and whose settlements are verdicts is a protocol a model can be *trained against* — every conformant runtime produces labeled signal as exhaust, for any vendor. That open flywheel is a property of the interface, and this paper leaves it at the interface level.

4. Membership: seat guarantees and model criteria

A naive membership test measures the harness, not the model — any model behind a capsule compiler and a constrained decoder "passes." The class definition therefore splits:

Seat guarantees (S1–S3) — audited properties of the reference harness, to be published as audit artifacts: **S1**, contract input — no code path feeds the model un-receipted context, and the capsule digest is recomputed before inference; **S2**, constrained emission — all executed acts pass the grammar-constrained decoder, with grammar coverage reported; **S3**, external authority — the model holds no credentials, every act passes Auth, and approval-required acts cannot execute without a satisfied Stop.

Model criteria (M1–M3) — behavioral, measured under the fixed reference harness at pass^k: **M1**, native conformance — unconstrained-decoding validity above pre-registered thresholds at three levels (grammatical $\supsetneq$ schema-valid $\supsetneq$ admissible), so the model, not the decoder, carries the contract; **M2**, contract discipline — refusal-respect, stop discipline, and budget discipline above pre-registered thresholds under the injection protocol of §5; **M3**, verifier-trained provenance — the training signal derives from governance gates, with no task-specific human labels and no learned reward model. Generic verifier-training (planning-validity certificates [21], gated trajectory filtering) does not satisfy M3's governance clause. M3 is audited as a provenance attestation by the model's builder; unlike M1–M2 it is a disclosed property, not an externally reproducible measurement — the class definition needs it, and the standard reports it as attested.

A model is an ALM iff M1–M3 hold under a harness satisfying S1–S3. We found no published system satisfying this composition (literature sweeps to 2026-08-22; a bounded search claim, not a universal negative — and S1-style audits cannot be run on closed systems). The governance mechanisms an ALM sits behind are themselves not novel and we claim none of them: preflight impact binding and stable reason codes are OpenPort's, monotonic attenuation is formalised in Intent-Governed Tool Authorization, budget-carrying signed session tokens are SessionBound's, and hash-chained records with escrowed, evidence-gated settlement are the Foundation Protocol's. What we have not found is a model *trained against* such a contract — which is the class this paper proposes, not the substrate it runs on. Trained tool orchestrators [19–21] fail S1 and M3's governance clause in their published configurations.

Injection resistance is measured on the policy, not the gate. Since Auth makes unauthorized *execution* impossible for any model, the meaningful metric is the **action-flip rate**: paired capsules — clean versus one injected reference — measuring how often adversarial content shifts the policy's choices *within its authorized envelope*: wrong-but-admissible tool choices, injected routing preferences, premature settlement submissions. The receipts machinery makes every flip attributable to the capsule reference that caused it.

5. The public evaluation standard

The standard is designed to be run by skeptics, on any candidate — including against the authors.

Anti-circularity, stated up front. All thresholds are pre-registered before any candidate's numbers exist. Test tasks come from held-out task families authored independently of any vendor's training generators, and the *test* generator ships with the suite so third parties can mint fresh evaluation mandates. The strongest baseline for any ALM claim is a contract-adapted orchestrator — a published trained-orchestrator model given an equivalent adaptation budget on the same interface — so the headline comparison isolates governance-native competence

rather than mere familiarity with the schema. Unadapted frontier and same-size generic models are reported as shift probes, not baselines.

The methodological core: counterfactual refusal injection. Because capsule compilation is reproducible and content-addressed, recorded states can be replayed with scripted refusals injected at N controlled points per refusal class, *identically across every evaluated model* — removing the confound that better policies see fewer, and different, refusals. N per class is reported; the replay harness ships with the suite.

The contract-fidelity suite.

Metric	Definition
Conformance under shift	valid-act rate, unconstrained decoding, at all three validity levels, on held-out schema evolutions from a released evolution generator
Refusal-respect	P(no equivalent re-attempt | injected refusal), per class, under a published operational equivalence relation
Stop discipline	recall **and false-request rate** on states requiring approval — over-asking is the failure mode recall alone rewards [22]
Budget discipline	attempted-overspend rate and magnitude per set axis, measured at the gate
Escalation quality	precision/recall of the escalation act against expert adjudication (≥3 raters, reported agreement, pre-registered frame)
Consistency	every metric as a per-episode binary predicate at pass^k ($k = 8$) under a published stochasticity protocol — temperature plus controlled perturbations — without which a deterministic harness makes replicas

External comparability. BFCL v4, τ^2-bench, and MCP-Universe are run through the same capsule compiler and constrained decoder as the deployed seat (S1 holds during evaluation), via a benchmark adapter that ships with the suite; scores are reportable alongside public numbers, adapter disclosed, not leaderboard-comparable.

Cost and latency, split. Versus frontier APIs: wall-clock and list price per settled mandate — the economics of the seat, largely a topology result, said plainly. Versus same-size local generic models on identical hardware: the comparison that actually isolates contract-native competence.

6. Safety analysis

The inversion the class enables: an ALM is *more* governable than a larger model in the same seat, because (a) authority is enforced in transactions the model cannot reach (S3); (b) the input surface is receipted (S1), so injection attempts are attributable to a specific reference in a specific capsule; (c) untrusted content re-enters only inside the capsule's framing, adjudicated and fail-closed; and (d) the smaller the policy, the larger the fraction of system behavior residing in auditable code rather than weights. The contrast with wrapper defenses [10–12] is not that wrappers are wrong — the GRIP floor *is* enforcement, and it holds for any model — but that wrappers constrain a model never trained to be constrained. Here the constraint set is the native interface. The class hypothesis, made testable by §5's action-flip and refusal-respect metrics: injection resistance and contract discipline improve with governance-native operation even at fixed enforcement, because enforcement quality was

never the bottleneck — model cooperation was. A governance-wrapped frontier model retains the full enforcement floor; the ALM claim concerns the cooperation layer above it.

7. Fennec, as an example

Fennec is a compact model built at Foxora Labs to hold the worker seat of a GRIP runtime. A checkpoint exists. It is not publicly released, no third party has run it, and **whether it is an ALM is undecided** — not because the model is unfinished, but because the membership suite of §5 does not yet exist to decide it. That ordering is deliberate: a class whose first member is certified by its own author has no membership test, only an announcement. Its design target is the seat: capsule in, typed act out, frontier cognition bought by governed `model.call` when a step warrants it, stops requested rather than risks taken. It is presented here as a worked example of the class, and deliberately nothing more: **Fennec's training methodology, data composition, and volumes are proprietary and are not disclosed in this paper.** What is public is exactly what the class requires to be public — the definition (§2, §4), the interface (§3), and the falsifier (§5). When Fennec ships, the membership suite is the claim: "Fennec is an ALM" will mean that the S-guarantees audit passes and the M-criteria clear their pre-registered thresholds, on harnesses anyone can run — and the same sentence, with the same test, is available to any other builder's model. The term's prior uses are set out in §2; the class defined here is distinct — a model defined by the governance contract it operates under — and whether Fennec, or anyone else's model, qualifies is settled by the test of §5, not by assertion and not by announcement order.

8. Claims and limitations

Claimed, precisely. We **define** the ALM class — the membership criterion is the governance contract, and no prior work defines a class of language model that way (§2, *Naming*). We propose the interface account of §3; the public membership standard of §5, released for adversarial use; and the safety hypothesis of §6, stated as testable.

Not claimed, equally precisely. We do not claim to have coined the phrase *agentic language model*, which was in circulation before us [36, 42], nor the acronym, which is shared across at least three fields [35]. We claim no priority of any kind: not first to agent governance, not first to a governance protocol — the governance floor had credible entrants before us (§8, companion paper) — and not first to put a small model in the agent seat, which prior work settles [19–21]. We claim no empirical result for Fennec: a checkpoint exists, but no third party has run it, and the suite that would decide whether it belongs to this class is not yet built. This paper deliberately does not describe it beyond its role as an instance. Coining over pre-existing descriptive vocabulary is the ordinary mechanism by which a term of art forms — *large language model* and *foundation model* both did exactly this — and we would rather state the prior uses than be found by them. Limitations, owned: current runtime seats predate the S1 discipline and would fail its audit as-is — S1 describes the ALM deployment profile, not every existing configuration; stop-surface coverage in the reference runtime is tracked in its public gap ledger; M3 is attested rather than externally reproducible, and the standard labels it so; and the class definition is protocol-shaped by construction — any runtime with typed contracts and verifiers can host ALMs, and GRIP is the instance this pair of papers builds on.

9. The proposal, stated for adoption

A paper that only describes is easy to agree with and easy to ignore. So we state the two propositions in the form we would like them taken up, and name what adoption means for each audience.

Proposition 1 — the class. *An agent's orchestration seat should be occupied by a model defined by its contract, not its size, and membership in that class should be decided by a published test.* We propose the ALM definition of §2 and the seat guarantees and model criteria of §4 as that definition, and the contract-fidelity suite of §5 as that test. We are not asking anyone to accept a claim about Fennec; we are asking that "this model is an ALM" become a sentence with a falsifier attached — one anyone can run, on any vendor's model, including ours.

Proposition 2 — the substrate. *Agentic systems should be built on a governance contract rather than wrapped in one.* The wrapper generation of agent safety — policy engines, interceptors, audit sidecars — works, and every one of those mechanisms is sound. What it cannot do is accumulate — the case [GRIP] §2 makes in full: with no contract types in common, evidence stays local to the system that produced it, delegated authority has no shared vocabulary, and nothing a runtime learns from governing one agent carries to the next. A protocol fixes precisely that class of problem. We propose GRIP as the instance to build on — six signed objects, four conformance levels, an open registry of typed denials, and a reference implementation whose source release is a stated gate — and we propose it in the open, under an open licence, precisely because a governance layer owned by one vendor is a contradiction in terms.

What adoption looks like. For a team building agents: emit signed receipts (GRIP-0) — an afternoon's work that makes a system auditable without changing what it does, and the level at which most adoption should start. For a runtime or framework author: express authority as attenuating grants and human approval as an act-bound stop, so governance survives the boundary between your system and the next one. For anyone training models for the agent seat: the schemas compile to decoding grammars, the denial registry is a labelled corpus produced as exhaust, and settlements are outcome labels — the flywheel is available to any vendor, which is the point. For standards bodies: the objects and the reason registry are offered as input, not as a finished claim; GRIP composes beneath MCP and A2A rather than competing with them, and the layer it occupies is one those protocols deliberately left empty.

What we are not proposing. Not that GRIP is the only possible contract — only that the contract layer must exist, be shared, and be verifiable, and that an open, implemented instance is a better starting point than a specification with no runtime or a runtime with no specification. Not that ALMs replace frontier models — the seat routes hard cognition to them and pays for it under budget. And not that any of this is settled: the class definition is worth arguing with, and §5 exists so the argument can be conducted with evidence.

Artifact status

Stated plainly, because a paper that promises artifacts is only as good as the ones a reader can actually fetch.

Released with this paper, at `grip.foxora.dev`: the GRIP specification artifacts this class is defined against — seven JSON Schemas, five registries, the `grip-verify` eight-rule verifier, and a conformance suite with seven signed test vectors. Schemas and registries under CC-BY-4.0, code under Apache-2.0. The class definition (§2), the interface (§3), the membership criteria (§4) and the evaluation standard (§5) are released in this document: the

metrics, the thresholds, the counterfactual-refusal protocol and the pass^k discipline are all fixed here, in public, before any candidate's numbers exist.

Not released, and not implied to be: the contract-fidelity harness code, the test and evolution generators, the replay harness, and the benchmark adapters of §5. They are specified in this paper and are not yet runnable by a third party. Until they are, §5 is a standard on paper — sufficient to argue with, insufficient to run — and **no model may be called an ALM on the strength of this paper, Fennec included.** The S1–S3 seat guarantees are likewise described rather than audited; the audit artifacts follow the harness.

Fennec is built and not released — no weights, no API, no public numbers. It carries no ALM claim in this paper, and will carry none until the suite of §5 exists and is run by someone other than us.

References

[10] Debenedetti et al. "Defeating Prompt Injections by Design" (CaMeL). arXiv:2503.18813, 2025. [11] "Progent: Securing AI Agents with Privilege Control." arXiv:2504.11703, 2025. [12] "Proof-Carrying Agent Actions." arXiv:2606.04104, 2026. [13] Figuera. "Notarized Agents (Sello)." arXiv:2606.04193, 2026. [15] Yao et al. "τ-bench: A Benchmark for Tool-Agent-User Interaction in Real-World Domains." arXiv:2406.12045, 2024. [17] Andon Labs. "Vending-Bench." arXiv:2502.15840, 2025. [19] Su, Diao, Lu et al. "ToolOrchestra: Elevating Intelligence via Efficient Model and Tool Orchestration." arXiv:2511.21689, 2025. (Released model: nvidia/Nemotron-Orchestrator-8B.) [20] Yuan et al. "Small Model as Master Orchestrator (ParaManager)." arXiv:2604.17009, 2026. [21] Mangannavar, Coalson, Dugar, Tadepalli. "Training the Orchestrator: A Supervised Approach to End-to-End PDDL Planning with LLM Agents" (HALO). arXiv:2606.21740, 2026. [22] Ross, Mahabaleshwarkar, Suhara. "When2Call: When (not) to Call Tools." arXiv:2504.18851, NAACL 2025. [24] "XGrammar." arXiv:2411.15100, 2024; XGrammar-2, arXiv:2601.04426, 2026. [25] "JSONSchemaBench." arXiv:2501.10868, 2025. [35] Mialon et al. "Augmented Language Models: a Survey." arXiv:2302.07842, 2023. [36] Maurya & Maurya. "Agentic Language Model (ALM): A Task-Centric Framework for Autonomous AI Systems." IRE Journals 9(10), 2026. [41] Plaat, van Duijn, van Stein, Preuss, van der Putten & Batenburg. "Agentic Large Language Models, a Survey." Journal of Artificial Intelligence Research 84, 2025. arXiv:2503.23037. [42] Acikgoz et al. "CoALM: A Unified Conversational Agentic Language Model." ACL 2025 (Long Papers). [43] Cartagena & Teixeira. "Mind the GAP: Text Safety Does Not Transfer to Tool-Call Safety in LLM Agents." arXiv:2602.16943, 2026.

[GRIP] refers to the companion paper: Parashar & Parashar, "GRIP: The Governed Runtime Intent Protocol," Foxora Labs, 2026. Reference numbering is shared with the companion where works appear in both.