Making Hidden Misalignment Observable: Interpretive Drift Detection for Latent Pragmatic Misalignment

Jacky Doll[1], Mei Si[2]

[1]Rensselaer Polytechnic Institute, Troy, New York, USA
[2]Rensselaer Polytechnic Institute, Troy, New York, USA

Abstract

Collaboration fails more often, not because someone is uninformed, but because people's interpretations of each other's messages quietly drift apart. In multi-turn work-planning, writing, coordination-small interpretation shifts can accumulate into *interpretive drift*: **User 1** and **User 2** proceed with different task semantics while the surface dialog remains locally coherent. We frame this failure mode as *latent pragmatic misalignment* and argue that it is difficult to debug because it leaves few "trouble" signals in transcripts until a downstream tool call, plan step, or external stakeholder makes the implicit assumption consequential.

To make hidden divergences observable, we propose *Interpretive Drift Detection (IDD)*, a lightweight, developer-facing observability layer for communication-support agents. IDD tracks a conservative schema of task-critical variables (e.g., term meaning/scope, deadlines, owners, authority, commitments) as *actor-indexed belief snapshots* and (crucially) *second-order snapshots* that approximate what **User 1** appears to assume about **User 2**' interpretation. When User 1's assumed values for User 2 diverge from User 2's expressed commitments and persist, IDD emits *typed discrepancy alerts* (semantic, temporal, resource, role/authority, commitment) and links them to minimal evidence (which variables disagree and the utterance spans that support each value), plus oversight hooks (clarify, confirm, pause/undo). We present concrete examples of the phenomenon and a worked case "Q3" illustrating how typed alerts can turn transcript archeology into a bounded search problem. Evaluation is a future work; we outline a human-centered program that measures both detection quality and developer utility.

Keywords

latent pragmatic misalignment, interpretive drift, common ground, shared knowledge, common knowledge, pragmatics, observability, developer tools, LLM agents, belief tracking

CEUR-WS.org/Vol-4266/paper04.pdf

1. Introduction

In collaborative work, people routinely coordinate through language that is underspecified: terms are polysemous ("Q3"), standards are relative ("early"), and responsibilities are negotiated implicitly ("approval"). Coordination succeeds when interlocutors form sufficiently aligned mental models of what the other person knows, intends, and will treat as binding. It fails when those mental models quietly diverge. Critically, such mismatches can remain *latent*: the interaction stays fluent and locally coherent, yet **User 1** and **User 2** proceed under different task semantics until a downstream action makes the implicit assumption consequential.

In this paper, a mental model is an interlocutor's task-specific coordination state: a compact representation of the work being done that includes the operative meaning/scope of key terms, the relevant time frame/deadlines, the owners and decision rights/authority, and the commitments/acceptance criteria that make an action "done." A second-order snapshot is a computational proxy for second-order mentalizing (belief attribution): at a given turn, it represents what agent A currently takes agent B to mean/assume about those coordination variables (A's model of B's interpretation), and it can be compared against B's expressed commitments. This is intentionally not a "ground truth" record; it is an alignment diagnostic for when dialogue remains locally fluent yet the operative assumptions for downstream action have diverged. We relate this to common ground as public evidence that an interpretation is mutually recognized (what was said, acknowledged, or recorded), and to common knowledge as the

Joint Proceedings of the ACM Intelligent User Interfaces (IUI) Workshops 2026, March 23-26, 2026, Paphos, Cyprus
dollj@rpi.edu (J. Doll); sim@rpi.edu (M. Si)
© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073
published 2026-06-27

stronger recursive mutual-recognition condition that reduces second-order uncertainty in coordination. IDD uses second-order snapshots to detect when a high-leverage assumption has not yet been stabilized as common ground, and then prompts repairs that turn it into an inspectable shared record—moving the team toward the practical benefits of common knowledge without requiring infinite recursion.

Grounding and repair accounts emphasize the detection of problems and the initialization of clarification [1, 2], but grounding is economical: people often stop negotiating meaning once the interaction seems "good enough" for progress [1, 3]. Latent pragmatic misalignment is especially likely for vague expressions dependent on context and interpretation that depend on implicit standards and comparison classes [4]. Psycholinguistic work further suggests that egocentric defaults and bounded perspective-taking can sustain divergent assumptions even during fluent interaction [5, 6, 7].

This paper frames *latent pragmatic misalignment* as a *human-human coordination problem* and proposes a role for LLM-based agents that is different from "the agent misunderstood the user." Instead, the agent operates as a *communication observability layer*: it monitors an ongoing conversation between two people and surfaces evidence that **User 1**'s working model of **User 2**'s interpretation is drifting away from what **User 2** is expressing (and optionally vice versa). Importantly, we do *not* assume that the agent must infer comprehensive user profiles. In many deployments, the work context already constrains what matters; IDD therefore tracks a conservative schema of coordination-critical variables (scope, deadlines, authority, commitments) and focuses on mismatches that can plausibly affect downstream actions. Here are two concrete human-human examples.

Temporal drift ("Q3")

User 1: As per our FY calendar, schedule the Q3 review with Finance. Keep it early.

User 2: Great. I'll put a 60-minute review on the calendar in September.

Drift: User 1 meant fiscal Q3 (May–July); User 2 used calendar Q3 (Jul–Sep).

Authority drift ("approval")

User 1: Send the draft once Alex approves.

User 2: Will do. I'll move ahead after Alex's review.

Drift: User 1 expected Pat's final sign-off; transcript looks coherent.

Practical impact. In everyday collaboration, interpretive drift can manifest as missed deadlines, rework, and misrouted approvals; in agent-mediated workflows it can also become *irreversible* (e.g., sending an email, publishing a document, committing code) before anyone notices. IDD is designed to reduce this downstream cost by surfacing a small number of high-leverage assumption diffs with evidence, so teams can repair meaning before it propagates into tools and artifacts. While we describe IDD as developer-facing, the same alert artifacts can be adapted for non-technical collaborators as lightweight "assumption checks" (confirm/override) embedded in the shared workspace.

Contributions. This workshop paper (1) characterizes *latent pragmatic misalignment* as a failure mode in human-human coordination and explains why it resists transcript-based debugging; (2) situates the phenomenon in theories of shared knowledge, common knowledge, and pragmatic signaling that underpin team comprehension and cohesion; (3) proposes *Interpretive Drift Detection (IDD)* - actor-indexed snapshots over a conservative schema plus *typed discrepancy alerts* make hidden drift observable by comparing User 1's inferred model of User 2 with User 2's expressed commitments; and (4) outlines oversight hooks and a human-centered evaluation plan, with developer utility as a primary outcome. IDD is intended as an instrumentation module that can be embedded in such systems *to support human teams*: it helps agent builders treat misalignment as a coordination problem between people, and provides artifacts (belief diffs, evidence spans, and repair hooks) that can be integrated into agent

Table 1
Typed drift categories and example task variables (conservative schema).

Type	Examples of variables likely to cause downstream failure
Semantic	definition/scope of key terms; polysemy; jargon-specific meanings
Temporal	calendar vs. fiscal references; time zones; offsets; deadlines
Resource	budgets/quotas; time/compute caps; "best effort" vs. reserved resources
Role/authority	decision rights; approval chain; who can execute/send/publish
Commitment	who promised what; owners; acceptance criteria; completion conditions

frameworks.

2. Phenomenon and problem formulation

2.1. Definition and distinction from classic grounding failure

We define **latent pragmatic misalignment** as a divergence in task-relevant interpretations or standards that remains *unrepaired* because interaction appears locally successful; it becomes visible only when downstream actions depend on the implicit assumption. Over multiple turns, this can accumulate into **interpretive drift**: gradual movement of task semantics without overt breakdown.

This differs from classic grounding failure, where misunderstandings surface as conversational trouble and trigger repair [1, 2]. In latent misalignment, surface interaction provides enough positive evidence to proceed, so repair does not occur even though the underlying interpretations differ [3]. For developers, the operational consequence is retrospective debugging: the transcript looks fine, yet a latent variable (deadline, authority, resource cap) only becomes visible when a plan step or tool call depends on it. In a grounding failure, the mismatch becomes conversational trouble and is repaired in situ:

User 1: Meet on Tuesday. **User 2:** Thursday works. **User 1:** No, Tuesday.

2.2. Why it stays hidden

Three mechanisms help explain why misalignment can persist: (1) *Economical grounding*: participants seek sufficient evidence to proceed, not perfect alignment [1, 3]. (2) *Locally sufficient agreements*: conceptual pacts and interactive alignment support efficient coordination without guaranteeing shared underlying representations [8, 9]. (3) *Egocentricity*: interlocutors often anchor on their own standards and adjust only partially for a partner's perspective [5, 6, 7].

In LLM-agent workflows, these mechanisms combine with tool execution and long-horizon plans: a quietly shifted assumption can propagate into calendars, tickets, code changes, or emails before it becomes salient.

2.3. Drift types that matter in agentic workflows

Not all semantics are equally important. We focus on variables that (a) commonly control downstream actions and (b) are often implicit. We keep the schema conservative (semantic scope, temporal anchors, resources, role/authority, commitments) because these are the classic coordination failure points in joint work. Teams most often break down when they mismatch on what the task counts as, who is responsible/authorized, and when commitments and deadlines bind. Table 1 summarizes the drift types we target.

2.4. Why agentic settings amplify the problem

LLM-agent benchmarks such as AgentBench and WebArena show that agents can produce locally plausible multi-step interactions, yet fail end-to-end objectives in interactive environments [10, 11]. Tool-use prompting approaches such as ReAct explicitly interleave reasoning and acting across steps, increasing the surface area where task semantics can silently shift over turns and tool calls [12]. These results motivate developer-facing observability primitives that can reveal hidden semantic drift even when dialogue remains fluent.

3. Theoretical framing: shared knowledge, common knowledge, and pragmatic signals

Our core claim is that latent pragmatic misalignment is best understood as a breakdown in how *coordination-relevant knowledge* is represented and signaled between partners during joint activity.

3.1. Shared knowledge vs. common knowledge in coordination

In coordination games, actors often need evidence not only about facts, but about the other actor's *expectations*. Formal analyses show why "almost common knowledge" can behave very differently from common knowledge [13]. Empirically, common knowledge plays a privileged role in enabling coordination because it avoids second-order uncertainty about whether *enough* shared knowledge has been achieved [14]. This distinction matters for agentic systems: tool calls and delegated actions make coordination "risky," so a hidden mismatch about *what we mean* can propagate until it becomes costly.

From shared to common ground. Latent pragmatic misalignment often persists when interaction produces only weak shared knowledge (both parties "heard" the same term) rather than robust common ground (public evidence that the interpretation is mutual). Targeted pragmatic signals, including parentheses, explicit acceptance criteria, and confirm-before-execute previews, can convert implicit assumptions into a shared record, reducing coordination risk.

3.2. Common-knowledge generators in human work and agent stacks

In human coordination, people rely on "public" events that make not only a fact, but its *mutuality*, easy to infer. Classic examples include open announcements, explicit acceptance, and actions performed in a jointly attended space. This matters because shared knowledge can create a second-order dilemma: even if I know that you know p, I may be unsure whether you will treat that as *sufficient* to coordinate, or whether you expect additional rounds of confirmation [14, 13].

Agentic systems introduce new candidate common-knowledge generators: tool previews shown to participants, confirmations that are logged in a shared workspace, and explicit "assumption cards" that persist across turns (e.g., "Q3=Fiscal, May-July, timezone=ET, final approver=Pat"). Quite the opposite, hidden system messages, private memories, and unobserved tool failures can produce the opposite: *private* commitments that one participant reasonably assumes are shared.

3.3. Pragmatic signals as evidence about meaning and mutual knowledge

Pragmatics emphasizes that meaning is inferred from context, goals, and expectations rather than decoded from literal content alone. In probabilistic pragmatics (e.g., Rational Speech Act models), listeners infer intended meaning by reasoning about a cooperative speaker, and speakers choose utterances to be informative given the listener's inferences [15, 16]. These models underscore why "locally coherent" dialogue can still harbor divergence: a listener can settle on one plausible inference while the speaker intended another.

3.4. Comprehension and cohesion in human-AI teams

Team cognition research frames coordination as supported by shared mental models (SMMs) and transactive memory systems (TMSs): teams perform better when members share expectations about tasks and roles, and know *who knows what* [17, 18]. Recent human-AI team studies similarly argue that AI can function as a knowledge source within a team's TMS, affecting how teams generate hypotheses and "speak up" [19]. From this perspective, latent pragmatic misalignment is not only a dialogue failure: it is a team-state failure that can reduce cohesion by increasing surprise and repair costs.

Where SMM/TMS context comes from in practice. IDD does not require reconstructing full team cognition, but it benefits from lightweight signals about shared artifacts and roles. In deployed settings, these cues can be retrieved from the same places teams already coordinate: calendars and time-zone settings, project plans and tickets, org charts and directories, policy/docs that encode approval chains, and prior decisions recorded in shared notes. An IDD layer can accept this material as optional context (retrieval-augmented "memory") and treat it as evidence with provenance alongside dialogue spans and tool metadata. When such context is absent or stale, IDD falls back to dialogue-only extraction and increases uncertainty, which in turn makes interventions more conservative (clarify/confirm earlier rather than assert drift).

4. Positioning: differentiating phenomenon, problem, and solution from prior work

4.1. Phenomenon: interpretive drift as pragmatic misalignment (not just error)

Work on conversational grounding describes how interlocutors build common ground through acknowledgement and repair [1, 3]. Recent analyses of language-model dialogue suggest that models often *presume* common ground rather than actively establishing it, producing measurable "grounding gaps" [20]. Latent pragmatic misalignment is the failure mode that results when this gap meets high-stakes tool use: the interaction can remain locally fluent while interlocutors commit to different task semantics.

4.2. Problem: why transcripts under-specify coordination state

In multi-step settings, each action can reify an implicit assumption into the world (a calendar event, a ticket, a code change). The transcript is only one slice of the team state; what matters is whether partners have aligned expectations (SMMs) and a workable account of who knows what (TMS). In human-AI teams, this is complicated by the agent's opacity and by the fact that "speaking up" about implicit assumptions is optional and can be interactionally costly [19].

4.3. Solution: from opaque failure to inspectable diffs

Prior debugging practice often resembles transcript archaeology. IDD proposes a different unit of analysis: typed diffs over coordination-critical variables, each tied to minimal evidence and paired with a repair move.

Unlike intent or slot-filling approaches that typically resolve to a single "best" interpretation for an individual user, IDD keeps *actor-indexed* interpretations explicit and focuses on persistent *cross-participant* divergence. This makes it complementary to interactive alignment techniques: where alignment smooths surface forms, IDD targets the small set of variables that gate downstream actions and therefore merit explicit grounding as a shared record.

Table 2
How our framing differs from adjacent problem framings in agent research.

Adjacent framing	How latent pragmatic misalignment differs
Hallucination / factuality	A drift episode can occur even when all facts are correct; the issue is mismatched *pragmatic commitments* (scope, authority, deadlines).
Model or data drift	The model need not change; the *interaction* drifts as partners settle on different interpretations.
Tool execution error	Tools may execute correctly; the wrong assumption was passed as an argument (e.g., wrong quarter system).
Instruction-following failure	Even with "correct" instructions, vague terms ("early," "important") require pragmatic inference and negotiation.

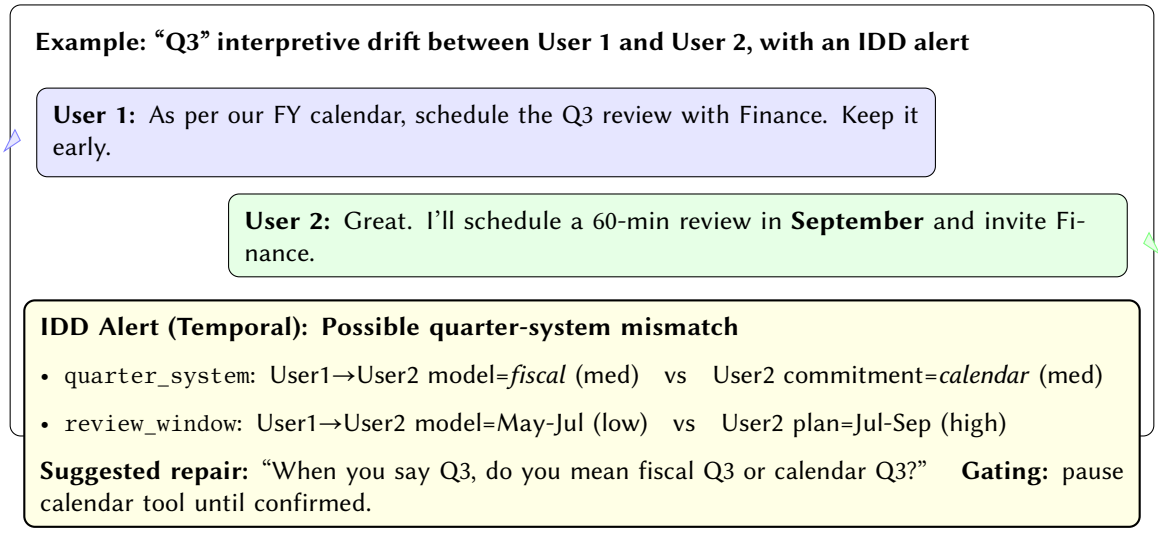

Figure 1: A conversation between two people can remain fluent while a coordination-relevant variable silently diverges. IDD surfaces the mismatch by comparing a second-order snapshot (User 1's apparent assumptions about User 2) to User 2's expressed plan, producing an interpretable diff with a targeted repair.

5. Design requirements: from theory to instrumentation

Theoretical accounts of grounding, common knowledge, and team cognition motivate concrete design requirements for drift-aware agents. We state six requirements that translate the phenomenon into implementable instrumentation choices.

5.1. R1: Track a small set of coordination-critical variables

Interpretive drift is most damaging when it affects variables that control downstream actions (deadlines, authority, scope, commitments). Rather than attempting to model "everything," drift-aware instrumentation should prioritize a conservative schema that covers these high-leverage variables (Table 1).

5.2. R2: Treat evidence as first-class (provenance spans and tool metadata)

Because misalignment can remain latent in fluent dialogue, a usable observability layer must attach disagreements to minimal, inspectable evidence: the utterance spans that support each competing value and the tool calls that consumed those values. This supports bounded inspection and reduces reliance on post-hoc guesswork.

Table 3
Design requirements mapped to concrete IDD features.

Requirement	IDD feature(s)
R1	Conservative schema of task variables; typed drift categories.
R2	Evidence spans for each field value; links to tool-call previews/arguments.
R3	Suggested repair templates; confirmation prompts; persistent "assumption cards."
R4	Persistence thresholds; risk-aware gating of high-stakes tools.
R5	Role/authority fields; commitment tracking; support for shared-workspace contexts.
R6	Evaluation plan targeting comprehension/cohesion outcomes, not only end-task success.

5.3. R3: Use pragmatic moves to strengthen common ground

If common knowledge is privileged for coordination [14, 13], then drift mitigation should emphasize pragmatic signals that create a public record of key assumptions: paraphrase-and-check, explicit acceptance criteria, and confirm-before-execute previews. Empirically, emphasizing common ground can improve outcomes in human-agent interaction (e.g., learning) [21].

5.4. R4: Make interventions risk-sensitive

Clarification has an interaction cost; letting drift persist has an action cost. Practical designs should modulate interventions by risk (e.g., outgoing emails, irreversible edits) and by uncertainty (confidence tags in snapshots). This aligns with human-AI interaction guidance that emphasizes user control and avoiding unnecessary interruptions [22].

5.5. R5: Support team cognition beyond the dyad

Many deployments involve multi-party workflows, shared documents, and policy constraints. Drift-aware designs should anticipate multi-party authority chains and shared mental model needs (who owns what, who approves what) [17, 18, 19].

5.6. R6: Measure "being on the same page" directly

Finally, drift-aware systems should be evaluated on outcomes beyond task completion: comprehension (predicting next actions), surprise and rework, and proxy measures of cohesion such as speaking-up behavior [19, 23].

6. Interpretive Drift Detection (IDD): a minimal observability layer

Because this is a workshop paper, we emphasize design clarity and integration points rather than full implementation details. IDD is a *developer-facing* layer that can be attached to an agent's interaction loop.

6.1. Conservative schema and agent-indexed belief snapshots

IDD tracks a developer-chosen set of coordination-critical variables (Table 1) as *actor-indexed belief snapshots*. For a focal dyad (User 1, User 2), the key design choice is to maintain a pair of aligned snapshots at each turn:

- z_t^2: what User 2 appears to commit to about the task variables at turn t (based on their utterances and context);

- $z_t^{1\to 2}$: a *second-order* snapshot approximating User 1's working model of User 2's interpretation of those same variables.

When available, the system can symmetrically track z_t^1 and $z_t^{2\to 1}$, but the minimal use case, and the framing emphasized in this paper, is diagnosing whether User 1 is acting under an inaccurate model of User 2.

This stance is compatible with mutual Theory-of-Mind modeling in that we treat belief as an attributed, actor-specific state rather than a single ground truth [24]. A practical instantiation can populate snapshots via a promptable extractor that outputs (i) a value/label per field and (ii) an uncertainty tag plus a supporting utterance span. The contribution here is the *observability interface* formed by snapshots, typed divergence, and repair hooks.

Each field f in a snapshot is stored as a tuple $\langle value, status, confidence, evidence\rangle$. We use *status* to distinguish explicit commitments (asserted/accepted), inferred assumptions, and unresolved uncertainty (unknown or multiple candidates). This operationalizes the difference between "User 1 assumes X" and "User 1 is uncertain between X and Y": low-confidence or multi-valued fields trigger clarification, while high-confidence mismatches are candidates for typed drift alerts.

Feasibility of second-order belief inference. Second-order attribution is challenging. Recent evaluations suggest that modern LLMs can succeed on some theory-of-mind and false-belief tasks, yet performance can be brittle to prompt and scenario variations [25, 26, 27, 28]. IDD therefore treats second-order snapshots as *hypotheses* rather than ground truth: we require (i) explicit evidence spans, (ii) persistence over turns, and (iii) risk-sensitive gating before any irreversible tool action. In this workshop contribution, we focus on the observability contract and repair hooks; robust second-order inference remains an open engineering and evaluation challenge.

6.2. Typed discrepancy alerts

We define divergence as a weighted sum of mismatched fields:

$$\Delta_t = \sum_{f\in\mathcal{F}} w_f \cdot \mathbb{1}\big[z_t^{1\to 2}[f] \neq z_t^2[f]\big],$$

and emit an alert when Δ_t exceeds a threshold and persists for k turns (to reduce noise). We map the highest-contributing fields to a typed alert (Semantic, Temporal, Resource, Role/Authority, Commitment).

Weights w_f are developer-controlled to reflect domain risk: e.g., authority and deadlines may be weighted higher than low-stakes phrasing differences. In an initial deployment, equal weights are a reasonable default; weights can then be tuned using incident logs and lightweight human feedback (e.g., "this alert mattered") or learned from outcomes such as rework and reversals. Here we treat weights as a configuration surface, consistent with IDD's role as an observability layer rather than a monolithic decision policy.

6.3. Worked example: temporal drift on "Q3"

A minimal snapshot comparison can reveal divergence even if the transcript contains no explicit contradiction. Table 4 shows simplified snapshots contrasting (i) User 1's apparent assumptions about User 2 (second-order) and (ii) User 2's expressed commitments.

Typed alert (Temporal). *Temporal drift:* `quarter_system` and `review_window` disagree while the intended action remains consistent ("schedule meeting").

Suggested repair and gating. A conservative repair is a targeted clarification: "When you say Q3, do you mean fiscal Q3 (May-Jul) or calendar Q3 (Jul-Sep)?" For high-stakes tools, the system can adopt *confirm-before-execute* gating: pause scheduling until confirmed.

Table 4
Example snapshot divergence for the "Q3" case (simplified).

Field f	User 1→User 2 model $z_t^{1\to 2}[f]$	User 2 commitment $z_t^2[f]$
`quarter_system`	fiscal (med)	calendar (med)
`review_window`	May-Jul (low)	Jul-Sep (high)
`tool_action`	schedule meeting (high)	schedule meeting (high)

What evidence supports fiscal vs. calendar? In Fig. 1, User 1 explicitly references an "FY calendar," which is a strong lexical cue that the quarter system is fiscal rather than calendar. More generally, IDD can treat fiscal-year terminology (FY, budget cycle), organization-specific calendars, and retrieved shared artifacts (e.g., a finance wiki page mapping fiscal quarters) as evidence. If evidence is weak, the snapshot marks `quarter_system` as uncertain rather than asserted, shifting the system toward asking the clarifying question instead of presenting a strong drift claim.

7. Developer workflow and oversight hooks

To be usable in real debugging, alerts must be actionable. We align the workflow with human-AI interaction guidance that emphasizes timely, specific, and user-controllable interventions [22]: detect a typed alert; inspect the minimal diff with provenance spans; repair via clarify/confirm; gate high-stakes tool calls (confirm-before-execute); and undo/replay when a wrong assumption has been operationalized.

Although we frame IDD as developer-facing, the same artifacts can be surfaced at different abstraction levels. In a builder console, alerts appear alongside traces and tool arguments; in end-user collaboration surfaces (chat, doc sidebar), an alert can be reduced to a one-sentence "assumption check" with a single click to confirm or override. Integration-wise, IDD can be implemented as middleware around an agent's message/tool bus: it observes turns, reads tool metadata (arguments and previews), writes snapshots to a store, and emits structured alerts to UI and logging systems without changing the agent's core planner.

8. Implementation sketch (for agent builders)

8.1. A minimal extraction prompt and update loop

We use a structured extractor that reads the latest turn and outputs a JSON object with (a) field values, (b) confidence, and (c) evidence spans. Importantly, the extractor is run twice: once to estimate User 2's expressed commitments and once to infer User 1's working model of User 2 (a second-order snapshot).

Listing 1: Simplified IDD update loop (pseudocode).

```
for each turn t:
  ctx = {dialogue up to t, tool logs, memory}
  z2[t] = extract_user2_commitments(ctx) # fields, values, confidence, evidence
  z1to2[t] = infer_user1_model_of_user2(ctx) # second-order snapshot

  diff = compare(z1to2[t], z2[t], weights)
  if diff.score > tau and persists_for_k_turns(diff):
     emit_alert(type=diff.top_type,
             fields=diff.top_fields,
             evidence=diff.evidence,
             suggested_repair=template(diff.top_type))

  if alert.type in gated_types and next_action_is_high_stakes():
     require_confirmation()
```

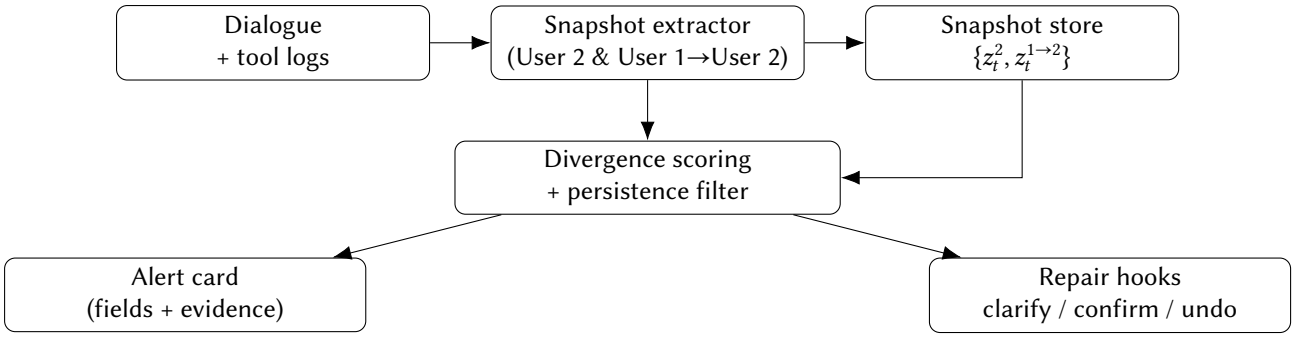

Figure 2: IDD as an attachable observability layer in an agent stack. It produces inspectable diffs and repair hooks rather than only end-to-end success/failure labels.

Extractor fallibility and self-diagnosis. The structured extractor used to populate snapshots can itself misinterpret pragmatics or overcommit. IDD mitigates this by treating extractor output as provisional, requiring evidence spans for each field, and running simple robustness checks (e.g., elicit alternative candidates; self-consistency across multiple samples) before emitting a high-confidence drift alert. In high-stakes cases, the preferred mitigation is not to "decide" correctly but to ask the minimal clarifying question that turns the variable into common ground.

8.2. UI patterns: evidence cards, not monolithic explanations

Rather than asking the system to produce a long explanation, IDD provides an *evidence card*: a typed alert, the disagreeing fields, and the spans that support each value. The goal is to increase comprehension and preserve cohesion by making assumptions explicit and repairable.

An evidence card also provides a concrete bridge from design requirements to interface elements (Table 3): it begins with a typed category badge (R1), shows field-level diffs with provenance spans and tool-call previews (R2), offers a one-sentence repair template plus a persistent assumption card to record the resolution (R3), and indicates risk/gating status for upcoming tool actions (R4). For multi-party settings (R5), cards can include affected roles/approvers; for evaluation (R6), they are logged as structured events.

9. Expanded examples: pragmatic signals that prevent drift

We provide two additional examples to illustrate how drift often centers on pragmatic, team-relevant variables. In each case, the system's role is to surface where User 1 appears to be assuming one interpretation for User 2 while User 2's responses support another.

9.1. Semantic drift: scope and acceptance criteria

User 1: "Make the onboarding doc more concise. Keep the important bits."
User 2: "Sure-I will cut the background section and keep the setup steps."
Risk: "important" can mean compliance requirements, technical prerequisites, or team norms. Here, the key variable is `acceptance_criteria`. User 1 may assume a compliance-centric criterion, while User 2 is optimizing for technical brevity. **Signal that helps:** a paraphrase that proposes an explicit split: "I plan to keep policy + setup sections and remove background. Is that what you consider the important bits?"

9.2. Authority drift: approval chains in multi-party work

User 1: "Send the draft once Alex approves."
User 2: "Alex said it looks good. I'll send it now."

Risk: approval policies can be implicit and context-dependent. IDD focuses on `approval_policy` and `decision_rights` and treats outgoing communication as confirm-before-execute when authority fields are uncertain.

10. Evaluation agenda: connecting detection to comprehension and cohesion

Beyond event-level detection metrics, we propose measuring outcomes that matter for human-AI teams.

10.1. Outcome measures

Comprehension: do users and developers correctly predict what the agent will do next (and why)? **Cohesion proxies:** perceived "being on the same page," reduced surprise, and reduced need for costly repair. Operationalizations include shared mental model similarity and speaking-up behavior [19, 23].

10.2. Study design sketch

We propose a controlled, multi-turn collaboration study with planted ambiguity triggers (quarter systems; vague thresholds; hidden approval policies) that allow dialog to remain locally fluent. Conditions compare (i) baseline traces (transcript + tool logs), (ii) ambiguity-threshold clarification prompts, (iii) first-order snapshot tracking only, and (iv) IDD with second-order snapshots, evidence cards, and targeted repairs.

A key comparison is (ii) vs. (iv): ambiguity-threshold strategies tend to ask generic questions whenever the agent is uncertain, which can be overly interruptive and still miss cases where one user confidently holds a wrong model of the other. By explicitly comparing $z_t^{1\to 2}$ to z_t^2 and requiring persistence, IDD can surface-targeted repairs only when a plausible mismatch is developing and is likely to matter for downstream action.

11. Implications and open questions

IDD suggests a shift in what we treat as a first-class artifact: not only plans and tool logs but also *coordination state*.

- **What should be "public" by default?** Which beliefs should be surfaced as a shared record (deadlines, definitions, owners)?
- **When should agents ask?** How can thresholds depend on tool risk and confidence?
- **What counts as evidence?** How should dialog spans, tool previews, and policy docs be weighted when updating snapshots?

12. Ethics and limitations

Belief snapshots and provenance spans may contain sensitive user information and inferred mental-state proxies. We recommend minimization, redaction, and strict access controls for logs and emphasize that alerts should remain advisory. A key limitation is that both first- and second-order snapshots are inferences: the extractor may misread pragmatics, and LLM-based mental-state attribution can be brittle across prompt and scenario variations. Consequently, IDD should privilege user-correctable repairs (clarify/confirm) and risk-sensitive gating over "silent" automatic resolution, especially when actions are irreversible or socially consequential.

13. Conclusion

Latent pragmatic misalignment is a subtle but consequential failure mode in multi-turn agentic systems: interaction can look successful while task semantics silently diverge. We contribute a phenomenon-focused framing grounded in shared vs common knowledge and pragmatic inference, and propose *Interpretive Drift Detection (IDD)*-typed discrepancy alerts over conservative, agent-indexed belief snapshots, to make such drift observable and repairable in developer and user workflows.

Declaration on Generative AI

During the preparation of this work, the author(s) used ChatGPT (GPT-5.2) in order to: draft and revise portions of the manuscript text (e.g., improving clarity and academic phrasing), perform grammar and spelling checks, and simulate peer-review feedback on framing and positioning. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the publication's content.

References

[1] H. H. Clark, S. E. Brennan, Grounding in communication, in: L. B. Resnick, J. M. Levine, S. D. Teasley (Eds.), Perspectives on Socially Shared Cognition, American Psychological Association, 1991.

[2] E. A. Schegloff, G. Jefferson, H. Sacks, The preference for self-correction in the organization of repair in conversation, Language 53 (1977) 361–382.

[3] H. H. Clark, Using Language, Cambridge University Press, 1996.

[4] C. Kennedy, Vagueness and grammar: The semantics of relative and absolute gradable adjectives, Linguistics and Philosophy 30 (2007) 1–45.

[5] B. Keysar, Communication and miscommunication: The role of egocentric processes, Intercultural Pragmatics 4 (2007) 71–84.

[6] B. Keysar, D. J. Barr, J. A. Balin, J. S. Brauner, Taking perspective in conversation: The role of mutual knowledge in comprehension, Psychological Science 11 (2000) 32–38. doi:10.1111/1467-9280.00211.

[7] W. S. Horton, B. Keysar, When do speakers take into account common ground?, Cognition 59 (1996) 91–117.

[8] S. E. Brennan, H. H. Clark, Conceptual pacts and lexical choice in conversation, Journal of Experimental Psychology: Learning, Memory, and Cognition 22 (1996) 1482–1493.

[9] M. J. Pickering, S. Garrod, Toward a mechanistic psychology of dialogue, Behavioral and Brain Sciences 27 (2004) 169–226.

[10] X. Liu, H. Yu, H. Zhang, et al., Agentbench: Evaluating llms as agents, 2023. URL: https://arxiv.org/abs/2308.03688. arXiv:2308.03688.

[11] S. Zhou, F. F. Xu, H. Zhu, et al., Webarena: A realistic web environment for building autonomous agents, 2023. URL: https://arxiv.org/abs/2307.13854. arXiv:2307.13854.

[12] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, Y. Cao, React: Synergizing reasoning and acting in language models, in: International Conference on Learning Representations (ICLR), 2023. URL: https://arxiv.org/abs/2210.03629.

[13] A. Rubinstein, The electronic mail game: Strategic behavior under "almost common knowledge", American Economic Review 79 (1989) 385–391. URL: https://ideas.repec.org/a/aea/aecrev/v79y1989i3p385-91.html.

[14] K. A. Thomas, P. DeScioli, O. S. Haque, S. Pinker, The psychology of coordination and common knowledge, Journal of Personality and Social Psychology 107 (2014) 657–676. doi:10.1037/a0037037.

[15] M. C. Frank, N. D. Goodman, Predicting pragmatic reasoning in language games, Science 336 (2012) 998. doi:10.1126/science.1218633.

[16] N. D. Goodman, M. C. Frank, Pragmatic language interpretation as probabilistic inference, Trends in Cognitive Sciences 20 (2016) 818–829. doi:10.1016/j.tics.2016.08.005.

[17] J. A. Cannon-Bowers, E. Salas, S. A. Converse, Shared mental models in expert team decision making, in: J. N. John Castellan (Ed.), Individual and Group Decision Making: Current Issues, Lawrence Erlbaum Associates, 1993, pp. 221–246.

[18] D. M. Wegner, Transactive memory: A contemporary analysis of the group mind, in: B. Mullen, G. R. Goethals (Eds.), Theories of Group Behavior, Springer, 1987, pp. 185–208. doi:10.1007/978-1-4612-4634-3_9.

[19] N. Bienefeld, M. Kolbe, G. Camen, D. Huser, P. K. Buehler, Human-ai teaming: Leveraging transactive memory and speaking up for enhanced team effectiveness, Frontiers in Psychology 14 (2023) 1208019. doi:10.3389/fpsyg.2023.1208019.

[20] O. Shaikh, K. Gligorić, A. Khetan, M. Gerstgrasser, D. Yang, D. Jurafsky, Grounding gaps in language model generations, 2023. URL: https://arxiv.org/abs/2311.09144. arXiv:2311.09144.

[21] A. Körner, A. Tolzin, A. Janson, J. M. Leimeister, R. Rummer, Common ground improves learning with conversational agents, Behaviour & Information Technology (2025). doi:10.1080/0144929X.2025.2541222.

[22] S. Amershi, D. Weld, M. Vorvoreanu, et al., Guidelines for human-ai interaction, in: Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems, 2019. doi:10.1145/3290605.3300233.

[23] E. Salas, D. E. Sims, C. S. Burke, Is there a "big five" in teamwork?, Small Group Research 36 (2005) 555–599. doi:10.1177/1046496405277134.

[24] N. C. Rabinowitz, F. Perbet, H. F. Song, C. Zhang, S. M. A. Eslami, M. Botvinick, Machine theory of mind, 2018. URL: https://arxiv.org/abs/1802.07740. arXiv:1802.07740.

[25] M. Kosinski, Evaluating large language models in theory of mind tasks, Proceedings of the National Academy of Sciences of the United States of America 121 (2024) e2405460121. doi:10.1073/pnas.2405460121.

[26] J. W. A. Strachan, D. Albergo, G. Borghini, O. Pansardi, E. Scaliti, S. Gupta, K. Saxena, A. Rufo, S. Panzeri, G. Manzi, M. S. A. Graziano, C. Becchio, Testing theory of mind in large language models and humans, Nature Human Behaviour 8 (2024) 1285–1295. doi:10.1038/s41562-024-01882-z.

[27] T. D. Ullman, Large language models fail on trivial alterations to theory-of-mind tasks, arXiv (2023). URL: https://arxiv.org/abs/2302.08399. arXiv:2302.08399.

[28] M. Sap, R. Le Bras, D. Fried, Y. Choi, Neural theory-of-mind? on the limits of social intelligence in large LMs, in: Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Abu Dhabi, United Arab Emirates, 2022, pp. 3762–3780. URL: https://aclanthology.org/2022.emnlp-main.248/. doi:10.18653/v1/2022.emnlp-main.248.