Agent Execution Delay Summary

Produced by: UNIC Evolve, May 4, 2026

1. Executive Summary

This report summarizes the May 4, 2026 reasons identified for slow or failed agent executions, using the user-provided incident facts associated with parent manager ticket `c1c69bc1-61c1-448b-be2e-1cac01233bed` as the authoritative source. The purpose is to separate confirmed operational facts from interpretation, expected long-running behavior, pathological failures, recommendations, and open questions without introducing unsupported details.

Three causes were identified:

1. **Playwright/Chromium `claudeuser` access delay:** Browser-dependent execution can become slow or blocked when access to the relevant Playwright/Chromium user context is delayed. This is treated as a pathological or failure-prone condition when it prevents expected agent progress. The available facts do not establish the underlying trigger, frequency, duration, or customer impact.
2. **Reasonable 4-hour manager QA loop:** The 4-hour manager QA loop is categorized as expected long-running behavior, not a failure by itself. Its operational risk is primarily interpretive: without clear status labeling, a normal QA wait can be mistaken for a stalled or failed execution.
3. **Claude-to-Codex reasoning mismatch at PR follow-up:** A reasoning mismatch at PR follow-up can cause handoff or continuation issues that slow or fail agent execution. The related reasoning fix is **in progress**. This report does not claim that the fix is complete or that the issue has been resolved.

The main operational conclusion is that the observed latency and failure reports should not be treated as a single failure class. One cause reflects expected review latency, while two reflect conditions that can block or degrade execution. Recommended follow-up should therefore focus on clearer state classification, better visibility into blocked browser access, and continued work on PR-follow-up reasoning alignment, while avoiding unsupported claims about metrics, incidents, root cause, or remediation completion.

2. Confirmed Facts and Source Boundaries

2.1 Confirmed Facts

The following facts are confirmed for this report:

- **Report date:** May 4, 2026.

- **Parent manager ticket:** `c1c69bc1-61c1-448b-be2e-1cac01233bed`.
- **Requested artifact path:** `ops/reports/agent-execution-diagnostics/2026-05-04-agent-execution-latency-failure-report.md`.
- **Authorized source of incident facts:** the user-provided incident facts in the manager prompt.
- **Research scope:** `research_rounds=1`, because this is an internal synthesis from supplied facts rather than an external deep-research request.
- **Required coverage:** the report must cover all three supplied causes:
  - Playwright/Chromium `claudeuser` access delay.
  - Reasonable 4-hour manager QA loop.
  - Claude-to-Codex reasoning mismatch at PR follow-up.
- **Required classification:** the report must separate confirmed facts, interpretation, expected long-running behavior, pathological failures, recommendations, and open questions.
- **PR-follow-up status constraint:** the Claude-to-Codex PR-follow-up reasoning fix must be described only as **in progress**.

2.2 Source Boundaries

This report is intentionally limited to the supplied facts. It does not include or infer:

- Secrets, credentials, tokens, private identifiers, or sensitive internal data.
- Raw logs, stack traces, console output, screenshots, transcripts, or unredacted operational records.
- Customer names, account details, user-level impact, support history, or customer-specific timelines.
- Invented incidents, metrics, durations, frequencies, severity levels, ticket IDs, owners, service names, or remediation dates.
- Unsupported root-cause claims, implementation details, rollout status, or claims that the PR-follow-up reasoning fix is complete.

2.3 Validation Boundary

The requested deliverable is a content artifact only. Validation is limited to confirming that the markdown file exists at `ops/reports/agent-execution-diagnostics/2026-05-04-agent-execution-latency-failure-report.md` and contains no placeholder text. No code execution, runtime validation, reproduction attempt, automated test run, browser test, or production verification is expected. No files should be created or modified outside `/ops/reports/agent-execution-diagnostics`.

3. Cause-by-Cause Diagnostic Interpretation

3.1 Playwright/Chromium `claudeuser` Access Delay

Confirmed fact: One identified contributor to slow or failed agent execution on May 4, 2026 was delayed access involving Playwright/Chromium and the `claudeuser` context.

Interpretation: When an execution depends on browser automation, delayed access to the expected browser runtime or user context can prevent the agent from reaching the work surface needed to proceed. This condition can present as latency if the agent is waiting for access, or as failure if the wait prevents the execution from completing successfully.

Operational significance: This cause should be treated as a blocking dependency issue when browser access is required for the task. The available facts do not identify the specific trigger, affected environment, frequency, duration, or remediation owner, so the report should avoid assigning a narrower root cause than the supplied facts support.

3.2 Reasonable 4-Hour Manager QA Loop

Confirmed fact: One identified reason for apparent execution latency was a manager QA loop lasting approximately 4 hours, and the supplied facts characterize this duration as reasonable.

Interpretation: The 4-hour manager QA loop is best understood as expected long-running behavior rather than a pathological failure. A QA or manager review loop can legitimately extend total elapsed time even when the underlying execution is progressing through the intended operational workflow.

Operational significance: The main issue is classification and communication. If a task is in a reasonable QA wait state, it should not be labeled or interpreted as failed solely because elapsed time is long. Clear status language is needed so stakeholders can distinguish "in review" from "blocked," "stalled," or "failed."

3.3 Claude-to-Codex Reasoning Mismatch at PR Follow-Up

Confirmed fact: A Claude-to-Codex reasoning mismatch at PR follow-up was identified as a cause of slow or failed agent execution. The related reasoning fix is in progress.

Interpretation: PR follow-up requires continuity between the reasoning used to understand a requested change and the execution behavior used to address that change. A mismatch at this handoff point can cause the follow-up agent to pursue an incomplete, inconsistent, or incorrect continuation path, increasing the likelihood of delay, rework, or failed completion.

Operational significance: This condition should be treated as a failure-prone handoff issue until the in-progress reasoning alignment work is completed and validated. The current evidence supports saying that a fix is in progress; it does not support saying the fix is complete, deployed, validated, or effective.

4. Expected Long-Running Behavior vs. Pathological Failures

The three identified causes should be handled as separate operational categories. Treating all elapsed time as failure would overstate the manager QA loop, while treating all delays as normal would understate blocking dependency and handoff issues.

Cause	Operational Category	Rationale	Handling Guidance

Cause	Operational Category	Rationale	Handling Guidance
Reasonable 4-hour manager QA loop	Expected long-running behavior	The supplied facts characterize the 4-hour QA loop as reasonable. Long elapsed time in this case reflects a review step rather than a confirmed defect.	Label as `in review` or equivalent. Do not classify as failed based on duration alone.
Playwright/Chromium `claudeuser` access delay	Pathological or failure-prone condition	Delayed access to a required browser context can prevent browser-dependent execution from progressing.	Label as `blocked` when access is required and unavailable. Track separately from normal review latency.
Claude-to-Codex reasoning mismatch at PR follow-up	Pathological or failure-prone condition	A mismatch at PR follow-up can disrupt continuation of work, causing delay, rework, or failed completion.	Treat as a handoff reliability issue while the reasoning fix remains in progress. Do not claim resolution until validated.

4.1 Expected Long-Running Behavior

Expected long-running behavior includes workflow time that is part of the intended operating model. The manager QA loop falls into this category when it is functioning as a review or quality gate. In this state, the execution may appear slow from an elapsed-time perspective, but the underlying condition is better described as waiting for review rather than blocked or failed.

The operational risk is misclassification. If the task state does not clearly show that review is underway, stakeholders may interpret normal QA latency as a stalled execution. Status language should therefore distinguish review waits from technical blockers.

4.2 Pathological or Failure-Prone Conditions

Pathological failures include conditions that prevent the agent from making expected progress, break continuity across execution stages, or require intervention before successful completion is likely.

The Playwright/Chromium `claudeuser` access delay belongs in this category when browser access is necessary for the task and the delay prevents progress. The available

facts support identifying access delay as a blocking dependency, but they do not support a more specific root-cause claim.

The Claude-to-Codex reasoning mismatch at PR follow-up also belongs in this category. PR follow-up depends on consistent reasoning across the handoff. When that continuity breaks, the execution can slow, take an incorrect continuation path, or fail to complete. The related fix remains in progress, so this condition should continue to be tracked as unresolved until completion and validation are confirmed.

4.3 Practical State Boundaries

The following state boundaries should guide interpretation:

- **Waiting:** execution is paused for an expected external step, with no evidence of a defect.
- **In review:** execution is within an expected QA or manager review loop, including the reasonable 4-hour manager QA loop described in the supplied facts.
- **Blocked:** execution cannot proceed because a required dependency, such as Playwright/Chromium `claudeuser` access, is unavailable or delayed.
- **Failed:** execution has reached an unsuccessful terminal state or cannot complete without corrective action.
- **In progress:** remediation work is underway, such as the PR-follow-up reasoning fix, but completion has not been established.

These distinctions should be preserved in operational reporting so that normal QA latency is not counted as failure, and genuine execution blockers are not hidden inside broad latency categories.

5. Recommendations, Open Questions, and Validation

5.1 Recommendations

- **Separate review latency from failure reporting:** Classify the reasonable 4-hour manager QA loop as an expected review state rather than a failed execution. Use clear status language such as `in review` so stakeholders understand that elapsed time is attributable to QA rather than a technical stall.

- **Expose browser-access dependency blocks clearly:** When Playwright/Chromium `claudeuser` access is delayed and browser access is required for the task, classify the execution as `blocked` rather than grouping it with generic latency. This will make dependency-related execution issues easier to triage without implying a root cause that has not been established.

- **Continue PR-follow-up reasoning alignment work:** Keep the Claude-to-Codex PR-follow-up reasoning fix tracked as **in progress** until completion and validation are confirmed. Reporting should avoid language implying that the issue has been resolved.

- **Use operational state labels consistently:** Distinguish `waiting`, `in review`, `blocked`, `failed`, and `in progress` states in execution summaries. This will reduce confusion between expected long-running behavior and pathological failure conditions.
- **Keep future diagnostics evidence-bounded:** Any follow-up report should continue to avoid unsupported metrics, customer-impact claims, remediation dates, owners, or root-cause details unless those facts are supplied by an authoritative source.

5.2 Open Questions

- What exact condition triggers the Playwright/Chromium `claudeuser` access delay?
- In which environments or execution paths does the `claudeuser` access delay occur?
- How often does the access delay result in a slow execution versus a failed execution?
- What status label should be shown to users or operators during a reasonable manager QA loop?
- Should the 4-hour manager QA loop have a communicated service expectation or review-time guideline?
- What evidence will be used to determine that the Claude-to-Codex PR-follow-up reasoning fix is complete?
- What validation will confirm that the PR-follow-up reasoning mismatch no longer causes slow or failed executions?
- Are there additional internal facts, outside the supplied prompt, that should be linked in a future version without exposing logs, secrets, or private data?

5.3 Validation

The content artifact should be validated as follows:

- Confirm that `ops/reports/agent-execution-diagnostics/2026-05-04-agent-execution-latency-failure-report.md` exists.
- Confirm that the artifact contains no placeholder text.
- Confirm that the artifact covers all three required causes:
  - Playwright/Chromium `claudeuser` access delay.
  - Reasonable 4-hour manager QA loop.
  - Claude-to-Codex reasoning mismatch at PR follow-up.
- Confirm that the report separates confirmed facts, interpretation, expected long-running behavior, pathological failures, recommendations, and open questions.
- Confirm that the report does not include secrets, raw logs, private data, invented incidents, unsupported metrics, additional ticket IDs, customer data, or unsupported root-cause claims.
- Confirm that the PR-follow-up reasoning fix is described only as **in progress**.
- Confirm that no files are created or modified outside `/ops/reports/agent-execution-diagnostics`.

No code execution, runtime validation, browser test, production verification, or reproduction attempt is expected for this content-only report.