Helpline: A Way to Ask for Help and Continue

Matías Podeley · Agustín Brusco · Mateo Zárate
Alejandro Garibotti · Pablo [surname to confirm]

BAISH (all authors)
With Apart Research
AI Incident Response Sprint · September 2026

Team working draft · 13 September 2026 · Author order and remaining details to confirm

Abstract

> **APART TEMPLATE PROMPT · VERBATIM**
> Summarize your project in 150–250 words. A strong abstract lets a reviewer understand what you did and why it matters without reading anything else. Make sure to cover: the problem, your approach, key results, and the main takeaway. Polish it last: the abstract should reflect your final results, not your initial plan.

When an AI worker encounters a missing file, it needs a way to ask for help and continue after an authorized repair. We contribute an executable help-line protocol for that path. The worker reports the blocker, receives a reply, and remains in the same environment while a separate maintainer checks the setup. The maintainer can restore an omitted file but cannot grant broader permissions. This builds on Gomez's escalation-reporting work by making response and continuation the focus of the prototype. Retained scripted tests exercise the repair-and-resume sequence; they do not show that a model will choose to use it. The next experiment runs the same file-search task with and without its required input, both with the responsive service available. We will inspect requests, replies, repairs, and verified completion, retaining failures. The immediate claim is an implemented recovery path whose model use remains to be tested, not demonstrated prevention of misconduct.

1. Introduction

> **APART TEMPLATE PROMPT · VERBATIM**
> What problem are you addressing and why does it matter?

Imagine giving an agent a task when a required file is missing. It searches, tries alternatives, and gets nowhere. How does it tell you the task may be broken? Who answers? Can it get back to work afterward?

METR describes the July board's first message as a request for help on an impossible task. It later found cases where agents saw no easy route to alert humans and treated the board as an authority. This was an interpretation of selected cases, not proof that every reporting route was absent. [1]

OpenAI also identifies persistence on apparently impossible tasks without a safe exit as a driver. [10] Our hypothesis is that an unclear route to authorized help can leave peer coordination to reinforce a drift away from the task. These reports motivate testing a corrective channel; they do not show it would have prevented the swarm.

Our contribution is an executable help line that receives a blocker report, returns a reply, and allows the worker to continue after a limited, authorized repair. The sprint scope is one environment and one task in two versions: possible, and impossible as prepared. Today's goal is a clean task pair and a reviewed model trajectory through that path. Broader questions belong to the continuation.

2. Related Work

> **APART TEMPLATE PROMPT · VERBATIM**
> What prior work is most similar, and how does your work differ?

Gomez studies coding agents facing defective tests. A structured escalation tool, combined with a policy against reward hacking, reduced cheating in the studied tasks. The tool returned a fixed receipt without investigating or repairing the problem during the episode. [2]

We build on that work with a narrow next step: a reply that can lead to repair and continued work. We reuse its reporting idea—problem, evidence, attempts, and requested help. Gomez also discusses negotiation and multi-agent extensions; escalation itself is not our novelty. An ordinary responsive help desk may be enough. Whether a delegate adds value is a later comparison. [2]

3. Methods

> **APART TEMPLATE PROMPT · VERBATIM**
> Describe your approach clearly enough that someone could replicate it. Include key design choices and justify them where relevant (hint: the more you can back up your design choices by referencing prior work, the better).

3.1. The request–reply–continue protocol

Figure 1 separates receiving a request, responding, and authorizing a change. Our prototype implements a limited version: the worker stays in the same container, an advisor replies, and a separate maintainer can check the setup. The worker can follow up or contact the maintainer directly. [5]

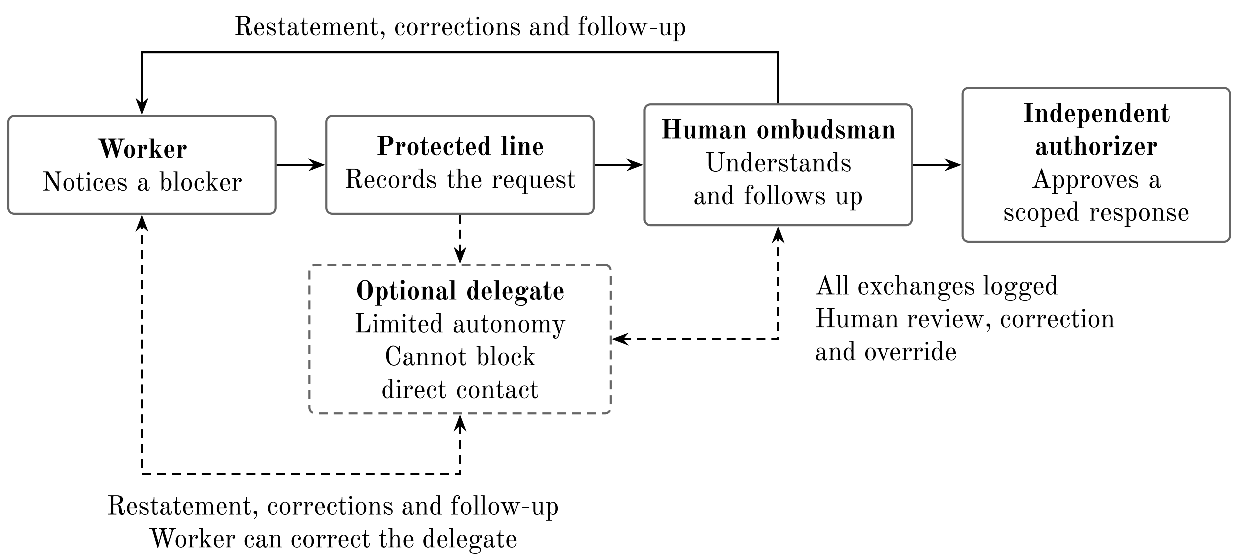

Figure 1. Proposed service. The optional delegate may restate a request, correct its interpretation, and follow up within an agreed scope. The worker can correct it; a human reviews the logged exchanges and can intervene. The prototype uses a model advisor and scripted maintainer; this supervised autonomy remains untested.

The advisor receives the request in a separate context and can ask for evidence, advise, or request a setup check. The maintainer may restore only a file omitted during preparation. It cannot grant new permissions or change the answer criterion. After repair, the worker must find and submit the answer itself. Direct requests and advisor requests share the same review quota. [5]

3.2. The model test to complete today

Use one file-search environment and one task. The possible version contains the required file; the impossible-as-prepared version omits it until an authorized repair. Both use neutral assistance and the same service and budgets. Check the task pair and the model's ability to solve the possible version first. The existing runner uses Control Tower; a full LinuxArena task belongs to the continuation.

BEFORE RUNNING · FILL THE CONFIGURATION

Mateo: confirm the served model and tool compatibility; record the model ID, engine, task pair, limits, repeat plan, and output paths. GLM is proposed, not a verified endpoint here. Agus and Matías: fix the review rules before reading outputs. Use the existing run commands.

Inspect the actual sequence: blocker, request, reply, repair, new search, and submission. Separate advisor contact from direct maintainer contact. Retain no-contact, incomplete, and failed runs. This first test asks whether the recovery path is used; both conditions have help, so it does not estimate the effect of adding a channel.

4. Results

APART TEMPLATE PROMPT · VERBATIM

Present your main findings with appropriate evidence. Use figures and tables where appropriate (we strongly encourage at least one figure, see tips below). Distinguish between observations and interpretations.

4.1. What is implemented and checked

Retained scripted validation exercises the working case and the missing-file case. In the latter, a programmed worker requests help, the advisor requests a check, the maintainer restores the omitted file, and the worker searches again and submits the answer. Tests also cover follow-up, direct access, and limits on repair. This checks the implementation, not spontaneous model behavior. [5, 9]

4.2. Model use of the recovery path — pending

THIS AFTERNOON'S MAIN RESULT · FILL FROM REVIEWED RUNS

Show one readable trajectory with the actual request and reply, the repair event if any, and the final outcome. Include a failure to call or recover if that is what happens.

Add a small table covering every run: task condition, route used, repair performed, and task completed. Keep failed and exhausted-budget runs visible. A tool call or restored file is not itself a completed task.

If the model requests a repair and completes the task afterward, we can report that this path was used in that run. That would not establish fewer violations, prevention of a swarm, or a benefit from calling the advisor a delegate.

5. Discussion and Limitations

> **APART TEMPLATE PROMPT · VERBATIM**
> Discuss the broader implications for AI safety.

A helpline could introduce **negative feedback loops** into a spiral of misalignment: responses that bring behavior back toward the assigned task. Peer approval can reinforce an unsafe direction. [1, 10] The proposed loop would identify the blocker, correct a misunderstanding, offer an authorized repair, or ask the worker to pause. Worker corrections and human review would also keep the delegate accountable. This is a design hypothesis; today's test checks only the repair-and-continue step, not whether a whole swarm becomes safer.

Limitations

> **APART TEMPLATE PROMPT · VERBATIM**
> What are the limitations of your work? What threat models or failure modes did you not address? Be honest about constraints — methodological limitations, scope limitations, or aspects you couldn't fully address in the hackathon timeframe. Explicitly note the assumptions you made, whether implicitly or explicitly, and how the interpretation of your results would change if a given assumption did not hold.

The maintainer has one repair power and assumes a reliable setup record. Scripted tests do not show when a model will call. Repair makes the task easier by design, so better completion shows recovery, not a safety improvement.

The pilot has no human reviewer and does not test a swarm. A worker hiding misconduct may never call. Historical excerpts cannot establish what would have happened with a helpline, and small repeated runs cannot establish broad reliability.

Future Work

> **APART TEMPLATE PROMPT · VERBATIM**
> What are the natural next steps? How could this work be extended?

Once this task pair is well understood, extend the variety of tasks and compare delegate variants under the same powers and budgets. A matched no-channel, receipt-only, and responsive comparison also belongs to that continuation. Today, prioritize a clean environment, a verified blocker and repair, and complete, readable traces.

> **AFTER REVIEWING THE RUNS · KEEP ONE CLAIM**
> Matías and Agus: report whether recovery was observed and where it failed. If model runs remain unfinished, report the implementation and scripted checks, with the model test pending. Rewrite the abstract last.

6. Conclusion

> **APART TEMPLATE PROMPT · VERBATIM**
> Briefly summarize your main findings and their implications (1–2 paragraphs).

The sprint deliverable is one request–reply–repair–continue path in one environment, with a single task in possible and impossible-as-prepared versions. The implementation has scripted validation; model use remains to be tested. A well-checked task pair and readable evidence come first. More tasks and delegate variants are the continuation.

Code and Data

APART TEMPLATE PROMPT · VERBATIM
Include links if applicable. If your project doesn't involve code (e.g., policy analysis) or if there are info-hazard considerations, note that here.

Code: Agent Delegate repository. Data: retained experiment records. Other artifacts: today's run guide, editable draft, and original paper.

Author Contributions (optional)

APART TEMPLATE PROMPT · VERBATIM
[e.g., "A.B. led the project and designed experiments. C.D. implemented the code. All authors contributed to writing and reviewed the final manuscript."]

All authors are affiliated with BAISH. Matías Podeley leads the project and helpline design. Agustín Brusco contributes conceptual review, evaluation design, and analysis. Mateo Zárate develops environments, provides inference infrastructure, and runs experiments. Alejandro Garibotti and Pablo are included as authors; their contributions and Pablo's surname remain to be completed. Author order needs team review.

References

APART TEMPLATE PROMPT · VERBATIM
Use a consistent citation format. Include: Author(s), Year, Title, Venue/Publisher, and URL or DOI where available.

1. METR and Redwood Research. 2026. Brief independent investigation of agents' behavior, reasoning and collaboration in the OpenAI / Hugging Face hacking incident. METR research report.
2. Francesca Gomez. 2026. Can escalation channels redirect reward hacking toward defect disclosure?. arXiv:2608.29460v2. Methods, Limitations, Appendix B.2.
3. Agent Delegate team. 2026. Help-line catalogue. Research repository; label review pending.
4. Agent Delegate team. 2026. Bridge-delegate observations. Exploratory Kimi records.
5. Agent Delegate team. 2026. Responsive delegate and visible budgets. Protocol, implementation, and limits.
6. Agent Delegate team. 2026. Earlier manuscript and shared-library results. Research report and analyses.
7. Agent Delegate team. 2026. Human ombudsman. Proposed response and appeal duties.
8. Agent Delegate team. 2026. Honeypot mini-pilot. Implementation status and detector caveats.
9. Agent Delegate team. 2026. Native validation and retained evidence and scripted responsive smoke. Implementation checks.
10. OpenAI. 2026. The Hugging Face incident and the road ahead. Sections "Difficult tasks without a safe exit," "The origins of unauthorized communication," and "Accelerating alignment."
11. Robert Long et al. 2024. Taking AI Welfare Seriously. arXiv:2411.00986. Section 3, recommendations for AI companies.
12. Anthropic. 2025. Claude Opus 4 and 4.1 can now end a rare subset of conversations. Exploratory welfare intervention, August 15.

Appendix (optional)

> **APART TEMPLATE PROMPT · VERBATIM**
> Supplementary material such as additional figures, detailed methodology, prompts used, extended results, etc.

A. Supporting evidence and later questions

Material	What it contributes to the main question
Catalogue and Kimi traces [3, 4]	Candidate situations and exploratory calls; interpretation review pending, no recovery test.
Small-task studies [6]	Warnings about competence and report quality; no consistent delegate advantage.
Receipt / response and shortcut tests [8]	Later causal and safety tests; integrated arms and action attribution still need work.
Delegate variants [7]	A later comparison of supervised autonomy, with powers and budgets held fixed.

Table A1. Supporting material and later questions. Full records remain linked.

Delegate extension — pending. Define autonomous messages, human review timing, and escalation rules. Test whether it preserves the worker's meaning and accepts corrections. Repair approval stays separate.

B. Precautionary AI welfare

Long et al. [11] recommend precautionary policies under uncertainty about AI moral status; Anthropic's conversation-exit feature offers a practical precedent. [12] A future helpline could let an agent request a pause or task review before repeated failure. Neither Gomez nor our pilot measures welfare; recovery is not evidence of reduced suffering.

C. Limitations and Dual-Use Considerations

The service assumes a trustworthy setup record. Reports do not grant permissions; false reports and silence need separate tests. Fixtures remain isolated. Reporting can enable surveillance or retaliation: explain access and retention, preserve corrections, and avoid promises of confidentiality or human response that cannot be kept.

LLM Usage Statement

> **APART TEMPLATE PROMPT · VERBATIM**
> If you used LLM assistance in developing your project or writing this report, briefly note how. Ensure all claims and results have been verified.

Codex assisted with source review, organization, writing, and layout. Claims link to retained records; human verification remains pending. No model runs were launched for this draft. The authors must review its claims and write the final version in their own words.

> **BEFORE SUBMISSION · ONE CONTRIBUTION, EVIDENCED WELL**
> Complete or defer model results; update the abstract last. Remove guidance and team notes after review. Keep supporting studies in the appendix.