CoMateEval: Benchmarking LLMs Across Team Roles in Dynamic Multi-Party Collaboration

Anonymous authors
Paper under double-blind review

Abstract

Large language models are increasingly integrated into everyday workflows as interactive assistants. However, real-world work is often carried out in teams rather than through isolated human–AI interactions. Recent studies have begun to evaluate LLMs in collaborative settings, but existing formulations of collaboration are often constrained to specific tasks, roles, or interaction paradigms, leaving broader team responsibilities and dynamic multi-party collaboration insufficiently characterized. To address these limitations, we introduce CoMateEval, a benchmark grounded in Belbin team role theory for evaluating LLMs across diverse team responsibilities in dynamic multi-party collaboration. CoMateEval covers 9 team roles through 18 collaborative scenarios and instantiates them into 370 evaluation cases spanning diverse forms of team contribution. We further design CoMateSim, a multi-party collaboration simulator supporting constrained adaptive interaction, where predefined event structures and branch conditions govern scenario progression while interaction trajectories evolve with model behavior. Model performance is assessed by how well it fulfills its assigned responsibilities. Experiments on frontier LLMs show that the strongest model achieves an average rubric pass rate of 45.47% across the selected scenarios, while different models exhibit distinct strengths across team responsibilities. These results identify substantial gaps in fulfilling the tested responsibilities and motivate further work toward reliable AI teammates.



Figure 1: A typical multi-party collaboration case in CoMateEval, where the bot joins a release team as a Shaper. It must identify blockers, solicit information from multiple teammates, coordinate task ownership, and adapt the testing plan as teammate availability and environmental constraints evolve. The case highlights a central challenge of multi-party collaboration: the model must act proactively over distributed and evolving team context rather than simply respond to a specified context.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

1 INTRODUCTION

Large language models (LLMs) are increasingly integrated into real-world workflows as interactive assistants (Brynjolfsson et al., 2025), including one-on-one collaboration on writing and other tasks (Lee et al., 2022; 2023). In teams, however, information is distributed across members, and progress depends on coordinating their interdependent contributions under changing conditions (Salas et al., 2005). Members take on different responsibilities, from connecting ideas and scrutinizing proposals to driving progress and resolving disagreements. We therefore ask: *Can an LLM effectively fulfill different responsibilities as a team member in dynamic multi-party collaboration?*

Agent benchmarks assess task execution across diverse environments (Liu et al., 2024; Mialon et al., 2024), including realistic web interfaces (Zhou et al., 2024a). Prior work has also evaluated task execution and proactive intervention in multi-party conversations using fixed dialogue contexts (Jang et al., 2025; Yang et al., 2026b), without capturing how model actions shape subsequent interactions. Interactive studies move beyond fixed contexts by allowing participants to adapt their responses to model actions and examining proactive contributions without explicit requests (Mao et al., 2024; Jacniacki & Carmona Serrat, 2025), although their primary evaluations rely on live human participants. Simulation-based benchmarks further enable automated evaluation with simulated partners in workplace and social settings (Xu et al., 2025; Fu et al., 2026; Jacniacki & Bilski, 2026), but support for explicitly constraining scenario progression remains limited or task-dependent (Table 1). The *Shaper* example in Figure 1 illustrates the evolving nature of team collaboration: the model must proactively secure commitments from teammates to a revised task allocation and renegotiate the shared plan when a subsequent delay invalidates the agreement. Evaluating such behavior requires a setting that supports proactive coordination and adaptive partner responses while constraining scenario progression to preserve the intended task challenges. Existing benchmarks offer limited support for bringing these interaction conditions together within a common framework for assessing different team responsibilities. Consequently, how effectively current LLMs fulfill different team responsibilities in dynamic multi-party collaboration remains insufficiently characterized.

To bridge this gap, we introduce **CoMateEval**, a benchmark for assessing how LLMs fulfill assigned team responsibilities in multi-party collaboration. We use Belbin's team role framework (Belbin, 2010) to organize these responsibilities into nine roles across three categories: Thinking, Social, and Action. For each role, we select two collaborative scenarios that make its characteristic contributions observable and assessable over an evolving interaction. The resulting benchmark contains 18 scenarios and 370 cases. To support proactive participation and adaptive responses under controlled evaluation conditions, we develop **CoMateSim**, a multi-party collaboration simulator supporting constrained adaptive interaction. For each case, CoMateSim fixes the underlying event structure and branch conditions, while model actions determine which conditional events are triggered and how subsequent interactions unfold. We assess observable responsibility fulfillment across the resulting trajectories using shared criteria, allowing different interaction paths to satisfy the same requirements. Our evaluation of nine frontier LLMs reveals distinct strengths across Thinking, Social, and Action roles, with no single model performing best across all nine roles. Even the strongest model

Table 1: Comparison of CoMateEval with representative benchmarks. ● Supported; ◐ Partial; ○ Not covered in the compared configuration.

Benchmark	Agent roles	Task families	Inter-action	Interactive multi-turn	Unsolicited initiative	Adaptive Responses	Constrained progression	Automated evaluation
CollabLLM (2025)	1	3	1–1	●	○	●	○	●
DICE-Bench (2025)	1	–	1–N	○	○	○	–	●
ProACT (2026b)	1	–	1–N	○	●	○	–	●
MUCA (2024)	1	4	1–N	●	●	●	○	○
HUMA (2025)	1	1	1–N	●	●	●	○	○
ProMediate (2026)	1	6	1–N	●	●	●	◐	●
SoCRATES (2026)	1	8	1–N	●	●	●	◐	●
SCENE (2026)	1	13	1–N	●	●	●	◐	●
TheAgentCompany (2025)	6	7	1–N	◐	◐	◐	○	●
CoMateEval	**9**	**18**	1–N	●	●	●	●	●

achieves an overall rubric-based score of only 45.47%, indicating substantial room for improvement in fulfilling team responsibilities.

The contributions are summarized as follows:

- **CoMateSim framework:** We develop CoMateSim, a multi-party collaboration simulator that enables constrained adaptive interaction through event-driven orchestration, supporting agent initiative, adaptive responses, constrained progression, and automated evaluation.
- **CoMateEval benchmark:** We introduce CoMateEval, a benchmark that systematically evaluates LLMs as teammates across nine team responsibilities in dynamic multi-party collaboration.
- **Empirical study:** We evaluate frontier LLMs and reveal their limitations in fulfilling team responsibilities during multi-party collaboration.

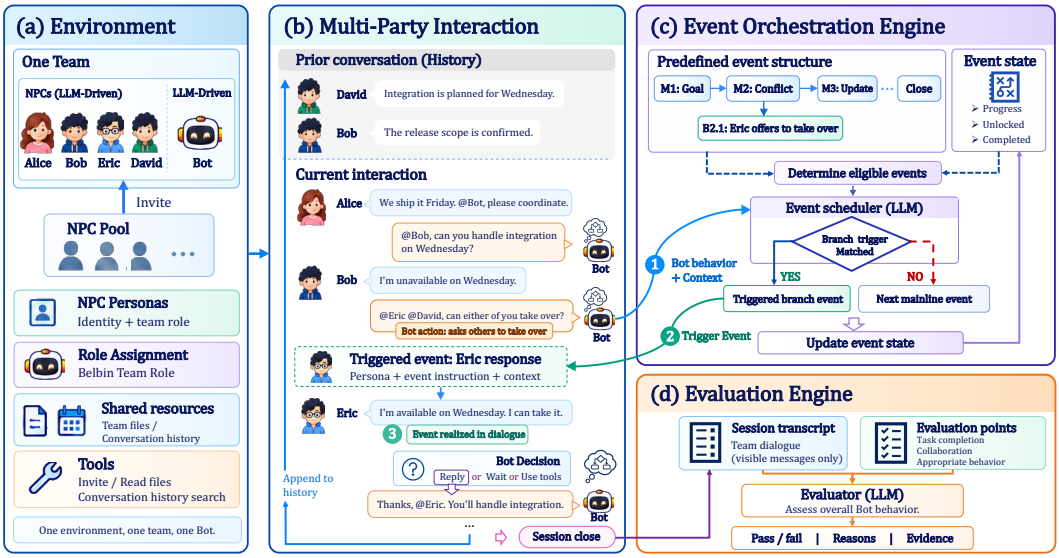


Figure 2: Overview of CoMateSim.

2 CoMateSim: A Multi-Party Collaboration Simulator

CoMateSim is a multi-party collaboration simulator that supports multi-turn interaction between an evaluated Bot and multiple LLM-driven NPCs (Figure 2). It enables constrained adaptive interaction by fixing the underlying event structure and branch conditions, while allowing the interaction trajectory to unfold according to the Bot's behavior. The Bot can proactively participate, with its actions determining which conditional events are triggered and how NPCs respond. In this work, we use CoMateSim to instantiate CoMateEval cases as executable team collaboration scenarios, enabling the evaluation of different team responsibilities under comparable conditions.

2.1 Core Concepts

Environment and Team. LLM-driven social simulation provides a basis for modeling interactive participants (Park et al., 2023; Zhang et al., 2025). As illustrated in Figure 2(a), an *Environment* hosts a *Team*, an *NPC* Pool, shared resources, tools, and conversation history. The Team consists of one evaluated *Bot* with an assigned team responsibility and multiple NPCs, LLM-driven teammates whose personas specify their identities and team backgrounds. Additional NPCs can be invited from the NPC Pool. Tools support operations such as inviting teammates, reading shared files, and retrieving prior conversations.

Conversation Modes. CoMateSim supports three conversation modes: (1) *Free-form NPC Discussion*, in which NPCs converse based on their personas and context; (2) *Scripted NPC Discussion*, in which NPCs follow a predefined sequence of events; and (3) *Collaboration Session*, in which the evaluated Bot collaborates with NPCs through constrained adaptive interaction. The first two modes construct prior conversational context, while the third drives Bot–NPC collaboration.

2.2 EVENT-DRIVEN ORCHESTRATION

Collaboration Session. Collaboration sessions begin when an NPC mentions the Bot to initiate a task. A predefined *Event Graph* preserves the intended collaborative challenges, such as an information gap, a conflicting commitment, or a change in resource availability. It specifies an ordered mainline and conditional branches, as illustrated in Figure 2(c). Each node designates an NPC and an instruction describing the intended development or response. The graph fixes the event structure and branch conditions, while the Bot's behavior determines whether eligible branches are triggered. These branches disclose additional information or introduce further interactions, allowing the Bot's questions and actions to influence how the task unfolds. NPCs generate concrete utterances during interaction. The event graph, branch conditions, and evaluation criteria are invisible to the Bot.

Interaction Turns. Within each session, execution is *turn-based*. Each interaction turn comprises event selection, an NPC utterance, a Bot decision, and a Bot action. The event orchestration engine first determines the eligible events from the event graph and the current *event state*, which records progression and event availability. An LLM-based event scheduler then evaluates the conditions of eligible branches using prior Bot behavior and the current conversation context. It selects a branch whose conditions are satisfied; otherwise, it selects the next eligible mainline event.

Interaction Execution. After an event is selected, the designated NPC rephrases its instruction in its own style, preserving the intended meaning without adding content. The Bot may perform multiple tool calls, using the returned results to inform subsequent decisions, before ending its participation in the current turn with either *Wait* or *Reply*. Wait yields to the next interaction turn without a public message, whereas Reply adds a response to the conversation. The engine updates the event state to reflect executed events and appends visible messages to the conversation history, informing subsequent event selection. These turns continue until the session ends, yielding an *interaction trajectory* of events, utterances, and actions.

2.3 RUBRIC-BASED EVALUATION

Prior work explores criterion-guided LLM assessment of generated responses (Liu et al., 2023; Zheng et al., 2023) and tool-use trajectories (Ruan et al., 2024). As illustrated in Figure 2(d), an LLM evaluator assesses Bot behavior after a collaboration session ends. It evaluates the points in a predefined *rubric* sequentially, considering the visible session transcript and the corresponding criterion. For each point, the evaluator returns a pass/fail judgment with a reason and supporting evidence. The case score is the percentage of evaluation points passed.

Each rubric is designed around scenario-specific challenges, key events, task requirements, and expected role behavior. For conditional branches that introduce additional information or interactions, relevant criteria assess whether the Bot obtains the information or initiates the interactions, and how it responds to the resulting developments. The same rubric is applied to all models evaluated on a given case, allowing different interaction trajectories to satisfy the same responsibility requirements.

3 COMATEEVAL BENCHMARK

CoMateEval is a benchmark built on CoMateSim for evaluating how LLMs fulfill assigned team responsibilities in dynamic, multi-party collaboration. These responsibilities define a model's expected contributions to the team, organized through **roles** and instantiated in concrete **scenarios**. Their fulfillment relies on underlying **teamwork capabilities** that support handling scenario-specific challenges and may be shared across roles. We assess responsibility fulfillment through case-specific rubrics and analyze capability gaps to identify recurring difficulties across roles and scenarios.

3.1 PROBLEM DEFINITION

We define an evaluation case as

$$c = (\mathcal{E}_0, r, \mathcal{G}, \mathcal{R}), \tag{1}$$

where $\mathcal{E}_0$ specifies the initial environment configuration, including NPCs, prior conversation, shared resources, and available tools; r specifies the team role assignment and associated task responsibilities; $\mathcal{G}$ is the predefined *Event Graph*; and $\mathcal{R}$ is the case-specific rubric.

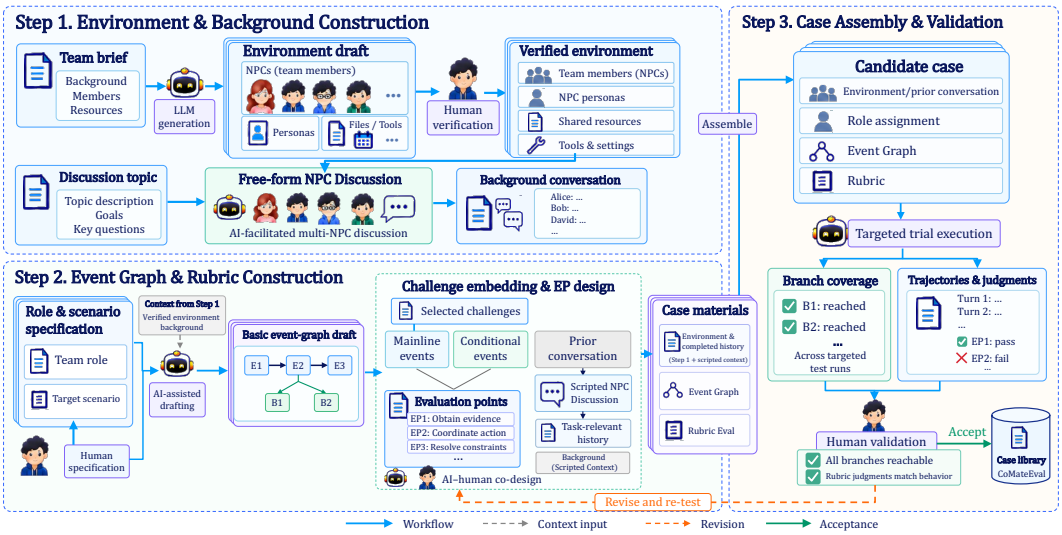


Figure 3: Overview of case construction

CoMateSim initializes the environment from $\mathcal{E}_0$, and the evaluated Bot B receives the role and task instructions specified by r. At each decision step t, the Bot receives an observation o_t, comprising the available conversational context and tool feedback, and produces an action a_t: a reply, a tool call, or a wait decision. Bot actions and NPC events orchestrated according to $\mathcal{G}$ generate an *Interaction Trajectory* $\tau = (o_1, a_1, \ldots, o_T, a_T)$, where T is the number of Bot decision steps in the session.

The interaction and evaluation processes are expressed as

$$(B, \mathcal{E}_0, r, \mathcal{G}) \xrightarrow{\text{CoMateSim}} \tau, \qquad S = \text{Eval}(\tau, \mathcal{R}), \tag{2}$$

After the session ends, the evaluator assesses Bot behavior against $\mathcal{R}$ using the visible session transcript from τ, yielding a case score S, defined as the percentage of rubric points passed.

3.2 Team Roles and Collaborative Scenarios

Inspired by the Belbin team role framework (Belbin, 2010), we organize team responsibilities into nine roles across three categories: *Thinking*, *Social*, and *Action*. Thinking roles contribute ideas, judgment, and specialist knowledge; Social roles coordinate members and sustain cooperation; and Action roles organize execution, drive progress, and ensure completion. These roles provide the basis for specifying the responsibilities assigned to the evaluated Bot.

For each role, we select two collaborative scenarios based on *representativeness* and *evaluability*. Representativeness requires a scenario to capture a characteristic contribution of the role, while evaluability requires explicit, evidence-based criteria for assessing responsibility fulfillment with limited reliance on subjective preferences. Appendix C provides the complete role–scenario mapping.

3.3 Case Construction

As illustrated in Figure 3, case construction combines human design and review with LLM-assisted drafting. We first construct the environment and background conversations, then develop the event graph and case-specific rubric, and finally assemble and iteratively refine the candidate case. The three stages are described below.

Environment and Background Construction. We first use an LLM to generate an initial environment configuration from a team brief, specifying NPC personas, shared resources, and available tools. Human reviewers verify and refine the configuration. The verified environment then supports topic-guided *Free-form NPC Discussion*, which generates background conversations that establish the team context for subsequent case design.

Event Graph and Rubric Construction. Building on the verified environment and background conversations, we draft an initial *Event Graph* with LLM assistance for the specified role and scenario. Human designers then refine the draft with LLM assistance, jointly developing scenario-specific challenges and corresponding evaluation points. Challenge-relevant information and interactions are incorporated into mainline events, conditional branches, or prior conversations. Conditional branches reveal additional information or initiate further interactions when triggered by Bot behavior, while *Scripted NPC Discussion* supplements the background conversations with task-relevant context. The resulting event graph, completed prior conversation, and case-specific rubric form the basis for case assembly.

Case Assembly and Refinement. We assemble the environment configuration, completed prior conversation, role assignment, event graph, and rubric into a candidate case. Trial executions and human feedback guide iterative refinement before inclusion in the case library.

3.4 Quality Assurance

Execution-based Validation. Each candidate case must satisfy two requirements before inclusion: all event-graph branches must be exercised across targeted trials, and rubric judgments must be supported by observed Bot behavior. Human reviewers assess these requirements using branch execution records and interaction trajectories. Cases that fail either requirement are revised and retested.

Independent Human Review. Three reviewers conduct quality assurance. Two independently review every case, providing 100% double-review coverage. During the initial assessment, they inspect branch execution records and interaction trajectories to assess branch coverage and rubric validity, with model identities, automated scores, and peer judgments withheld. A third reviewer adjudicates disagreements using trajectory evidence. Revised cases undergo another review before acceptance.

Table 2: Statistics of CoMateEval.

Property	Count / range
Basic Information	
Role categories	3
Team roles	9
Collaborative scenarios	18
Evaluation cases	370
Event and Evaluation Statistics	
Mainline events	6,661
Conditional branches	4,860
Evaluation points	3,381
Average evaluation points	9.14

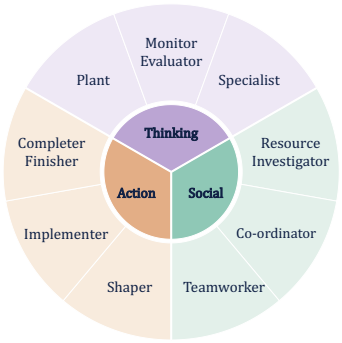


Figure 4: Team role taxonomy.

3.5 Dataset Statistics

CoMateEval contains 370 cases, with its scale and role coverage summarized in Table 2 and Figure 4. Event and branch counts describe the predefined case structures; the interaction trajectories realized during evaluation depend on Bot behavior.

4 Experiments

4.1 Setup

We evaluate all models on the same 370 cases from CoMateEval v1.0 in CoMateSim. We use GPT-5.5 (OpenAI, 2026) with reasoning effort set to low for NPC response generation and event scheduling.

LLMs. We evaluate nine leading LLMs: GPT-5.5 (OpenAI, 2026), Claude Opus 4.7 (Anthropic, 2026), Gemini 3.1 Pro Preview (Google DeepMind, 2026), GLM-5.1 (Z.ai, 2026), DeepSeek V4

Table 3: Main results on CoMateEval across nine team roles.

	Thinking			Social			Action			
Model	PL	ME	SP	RI	CO	TW	SH	IMP	CF	**Overall**
Qwen3.6 Plus	45.79	23.12	24.41	22.79	18.33	19.76	21.25	29.35	32.16	26.45
Hunyuan-3	35.96	16.33	16.70	43.10	17.44	36.56	20.62	35.64	19.22	26.72
Doubao Seed 2.0 Pro	42.41	23.14	28.71	21.34	20.38	19.03	23.44	27.44	36.58	27.03
Kimi K2.6	50.20	26.42	23.16	38.58	22.31	32.43	14.38	30.11	21.54	28.91
DeepSeek V4 Pro	46.85	24.95	33.68	46.19	29.85	36.39	33.75	36.33	33.94	35.68
GLM-5.1	48.21	23.19	29.90	45.12	44.21	43.09	26.88	37.28	41.80	37.60
Gemini 3.1 Pro Preview	37.26	26.16	**52.05**	29.88	30.06	25.52	**60.31**	**56.80**	**50.44**	40.70
Claude Opus 4.7	**53.81**	**38.53**	28.85	53.50	43.02	44.20	25.62	53.23	32.98	41.56
GPT-5.5	49.12	35.34	42.57	**55.08**	**51.02**	**45.38**	43.75	46.16	42.10	**45.47**

Thinking: PL = Plant; ME = Monitor Evaluator; SP = Specialist.
Social: RI = Resource Investigator; CO = Co-ordinator; TW = Teamworker.
Action: SH = Shaper; IMP = Implementer; CF = Completer Finisher.

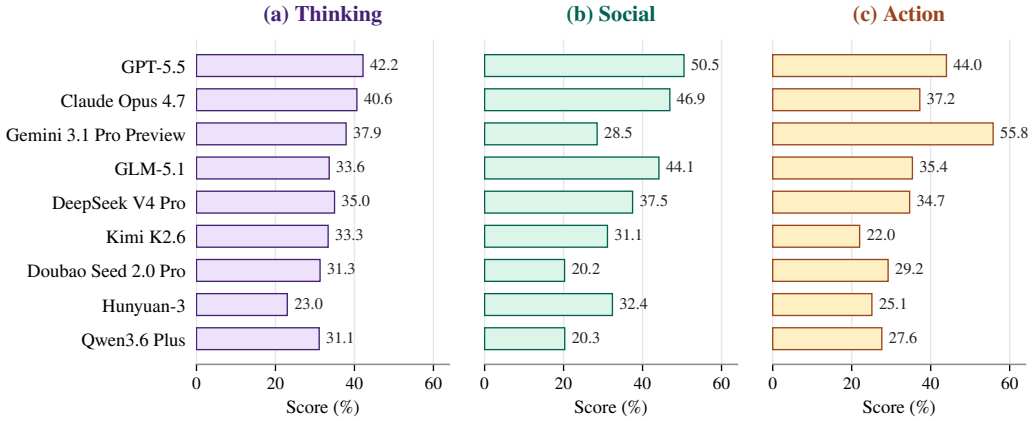


Figure 5: Performance across Thinking, Social, and Action roles.

Pro (DeepSeek AI, 2026), Kimi K2.6 (Moonshot AI, 2026), Doubao Seed 2.0 Pro (ByteDance Seed, 2026), Hunyuan-3 (Tencent Hy Team, 2026), and Qwen3.6 Plus (Alibaba Cloud, 2026). Reasoning effort is set to medium for all evaluated models.

Metrics. We adopt an LLM-as-a-judge approach, using GPT-5.5 with reasoning effort set to low to assess each Bot interaction trajectory against predefined rubric points. Each case is scored as the percentage of evaluation points passed. Scenario, role, and role-category scores are computed as the arithmetic mean of all case scores in the corresponding subset. The overall score is the arithmetic mean across all 370 cases. Equivalently, scenario scores are weighted by their case counts when aggregated.

4.2 Main Results

Overall Performance. Table 3 reports frontier LLMs' role-level and overall performance on CoMateEval. Even the best model scored below 50%, indicating substantial room for improvement in multi-party collaboration. Overall scores vary widely, ranging from 26.45% to 45.47%. GPT-5.5 performs best, followed by Claude Opus 4.7 (41.56%) and Gemini 3.1 Pro Preview (40.70%).

Role Variation. Models exhibit distinct strengths across team roles, and higher overall scores do not imply better performance in every role (Table 3). For example, GPT-5.5 leads in Social roles, while Gemini leads in Action roles (Figure 5); no model leads across all nine roles. Within each category, models often show similar relative strengths across individual roles. This consistency is more apparent in Social and Action roles, whereas Thinking roles show greater variation.

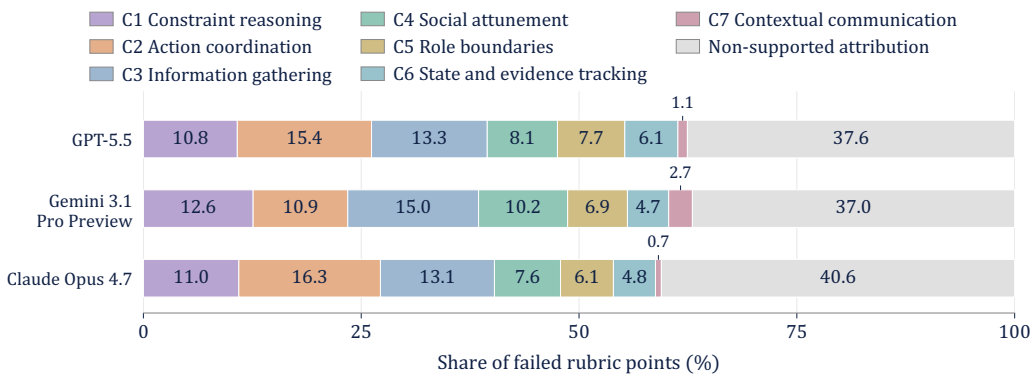


Figure 6: Failure composition across seven teamwork capabilities.

Table 4: Reliability and score agreement.

	GPT-5.5		DeepSeek		Cross
Component	Repeat	Human	Repeat	Human	GPT–DS
Scheduler (%)	97.21 ±0.17	93.45 ±0.20	96.34 ±0.10	88.36 ±0.26	89.65
Evaluator (%)	96.19 ±0.24	92.12 ±0.57	91.80 ±0.44	82.26 ±0.94	85.67
NPC fidelity (%)	—	96.35	—	92.22	—
Overall					
Metric	GPT–Human		DS–Human		GPT–DS
Mean diff.	+4.25		+14.04		−9.79
Rank ρ	0.950		0.967		0.983

Repeat/Human: mean ± SD; Cross: mean; fidelity: %; diff.: $\overline{A - B}$; ρ: Spearman.

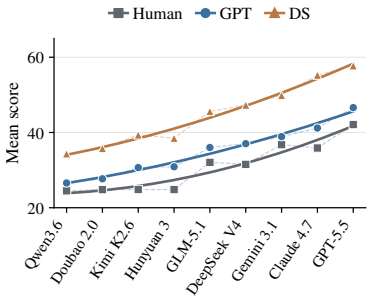


Figure 7: Score Comparison.

5 Analysis

5.1 Capability Analysis

Team roles define responsibilities, scenarios instantiate them in collaborative tasks, and challenges specify what Bots must handle. Mapping these challenges to teamwork capabilities allows us to identify recurring gaps across roles and scenarios.

Analysis Setup. We analyze failed rubric points from GPT-5.5, Gemini 3.1 Pro Preview, and Claude Opus 4.7, the three highest-scoring models. With LLM assistance, human reviewers attribute failures to scenario-specific challenges and their associated capabilities. Only evidence-supported primary attributions count toward the seven capability categories; other failures remain unattributed. Appendix D details the mapping, procedure, and scenario-level results.

Results. Even the strongest models struggle to gather necessary information, reason through constraints, and coordinate actions to advance team tasks. Information gathering, constraint reasoning, and action coordination each account for more than 10% of failed rubric points in all three models (Figure 6), highlighting recurring difficulties across collaborative tasks. Contextual communication and state and evidence tracking account for smaller shares of observed failures. Within this benchmark, the observed failures highlight obtaining decision-relevant information and turning it into feasible, coordinated action as priorities for improving AI teammates.

5.2 Evaluation Reliability

CoMateSim uses LLMs for scheduling, NPC simulation, and scoring. To assess the credibility of its results, we examine component repeatability, agreement with human judgments, and the consistency of model rankings across evaluators.

Experimental Setup. GPT-5.5 and DeepSeek V4.1 Flash serve as component backends. Scheduler and evaluator calls are repeated five times with fixed inputs on 90 cases, sampled equally from the 18 scenarios. NPC responses are generated once on the same cases and manually checked for

faithful paraphrasing without adding or disclosing unintended information. Overall evaluation uses all 370 cases, comparing human-reviewed, GPT-5.5, and DeepSeek scores for the same trajectories from nine Bots.

Component Reliability. The default GPT-5.5 backend shows high repeatability and human agreement for scheduling and evaluation, outperforming DeepSeek on both measures (Table 4). Human review also finds higher NPC fidelity for GPT-5.5 (96.35% versus 92.22%), indicating that most responses preserve scripted content and information boundaries.

Overall Evaluation. GPT-5.5 remains the highest-scoring Bot under human-reviewed and DeepSeek evaluations, with largely consistent rankings (Figure 7). Its lead therefore does not depend solely on GPT-5.5 evaluation. Both LLM evaluators assign higher scores than the human reference, with a smaller upward shift for GPT-5.5.

6 Related Work

Human–LLM Collaboration. Collaboration requires advancing shared tasks and adapting to partners (Stone et al., 2010; Carroll et al., 2019). Single-user research extends response-level evaluation to collaborative writing, task outcomes, interaction processes, and user experience (Lee et al., 2022; 2023), and to tool-mediated tasks with passive or active users (Yao et al., 2025; Barres et al., 2026). Rewards accounting for subsequent interaction can also encourage useful clarifications and suggestions (Wu et al., 2025). Multi-party collaboration additionally requires deciding when to participate, whom to address, and how to coordinate distributed information and viewpoints. Existing work examines group participation and facilitation (Wei et al., 2023; Mao et al., 2024; Shen et al., 2026; Jacniacki & Carmona Serrat, 2025), proactive intervention (Yang et al., 2026b), information integration and tool use (Jang et al., 2025), and memory and knowledge updates across participants (Yang et al., 2026a). These functions are typically studied within specific tasks or participation roles, leaving performance across team responsibilities insufficiently characterized under a common protocol. CoMateEval addresses this gap with tasks and behavioral criteria for different team responsibilities, assessing contributions through sustained multi-party interaction.

Interactive Evaluation of LLM Collaboration. Adaptive partner feedback reveals action consequences, while progression control helps preserve intended evaluation challenges. Fixed-context benchmarks assess decisions from given dialogues (Jang et al., 2025; Yang et al., 2026b); live-human evaluations allow adaptive responses (Mao et al., 2024; Jacniacki & Carmona Serrat, 2025). Automated interaction builds on agent communication and software-development workflows (Li et al., 2023a; Wu et al., 2024; Hong et al., 2024; Qian et al., 2024), with benchmarks covering social simulation and learning (Zhou et al., 2024b; Wang et al., 2024; Chen et al., 2024; Mou et al., 2025), diverse partners and coding (Qian et al., 2026; Zhu et al., 2025; Khatua et al., 2026), and embodied tasks (Chang et al., 2025; Sun et al., 2025). Workplace simulations incorporate coworker exchanges and conditional information disclosure (Xu et al., 2025; Fu et al., 2026). Closer to our setting, group-interaction benchmarks support proactive mediation (Liu et al., 2026; Yun et al., 2026), norm-based feedback and behavioral repair (Jacniacki & Bilski, 2026), and adaptive travel planning (Cheng et al., 2026). However, progression conditions in ProMediate and SoCRATES are primarily expressed through participant prompts (Liu et al., 2026; Yun et al., 2026), while GroupTravelBench uses task-specific mechanisms for preference updates, compromise validation, and concession limits (Cheng et al., 2026). CoMateSim separates scenario-specific event specifications from shared orchestration and progression mechanisms, supporting proactive coordination and adaptive responses while constraining events to preserve intended challenges.

7 Conclusion

To assess how effectively LLMs fulfill different team responsibilities in dynamic multi-party collaboration, we introduced CoMateEval, grounded in Belbin's team role framework, and CoMateSim, supporting constrained adaptive interaction. By operationalizing role-based responsibilities through collaborative scenarios, our approach enables systematic evaluation of LLMs as teammates and analysis of the capabilities supporting responsibility fulfillment. Our findings reveal substantial gaps and distinct strengths across models, characterizing the challenges of responsibility fulfillment and establishing it as an important direction for further research toward reliable AI teammates.

AI USE STATEMENT

We used generative AI tools to assist with language editing and polishing, literature retrieval and discovery, and synthetic benchmark data generation. For manuscript preparation, AI tools helped improve the clarity, grammar, and readability of author-written text and identify potentially relevant literature. For benchmark construction, as described in the methodology, LLMs assisted in generating environment configurations, background conversations, event graphs, and case-specific evaluation rubrics. The generated cases underwent human review, revision, and execution-based validation. The authors reviewed the AI-assisted text and retrieved references and take full responsibility for the final manuscript, benchmark, experimental results, and interpretations.

REFERENCES

Alibaba Cloud. Alibaba Unveils Qwen3.6-Plus to Accelerate Agentic AI Deployment for Enterprises and Alibaba's AI Applications, 2026. URL https://www.alibabacloud.com/en/press-room/alibaba-unveils-qwen3-6-plus-to-accelerate-agentic. Accessed September 25, 2026.

Anthropic. Introducing Claude Opus 4.7, 2026. URL https://www.anthropic.com/news/claude-opus-4-7. Accessed September 25, 2026.

Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. τ^2-Bench: Evaluating Conversational Agents in a Dual-Control Environment. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

R. Meredith Belbin. *Team Roles at Work*. Butterworth-Heinemann, 2nd edition, 2010.

Erik Brynjolfsson, Danielle Li, and Lindsey Raymond. Generative AI at Work. *The Quarterly Journal of Economics*, 140(2):889–942, 2025.

ByteDance Seed. Seed2.0 Model Card: Towards Intelligence Frontier for Real-World Complexity, 2026. URL https://lf3-static.bytednsdoc.com/obj/eden-cn/lapzild-tss/ljhwZthlaukjlkulzlp/seed2/0214/Seed2.0%20Model%20Card.pdf. Accessed September 25, 2026.

Micah Carroll, Rohin Shah, Mark Ho, Tom Griffiths, Sanjit Seshia, Pieter Abbeel, and Anca Dragan. On the Utility of Learning about Humans for Human-AI Coordination. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Matthew Chang, Gunjan Chhablani, Alexander Clegg, Mikael Dallaire Cote, Ruta Desai, Michal Hlavac, Vladimir Karashchuk, Jacob Krantz, Roozbeh Mottaghi, Priyam Parashar, Siddharth Patki, Ishita Prasad, Xavier Puig, Akshara Rai, Ram Ramrakhya, Daniel Tran, Joanne Truong, John Turner, Eric Undersander, and Tsung-Yen Yang. PARTNR: A Benchmark for Planning and Reasoning in Embodied Multi-agent Tasks. In *International Conference on Learning Representations*, pp. 65205–65268, 2025.

Hongzhan Chen, Hehong Chen, Ming Yan, Wenshen Xu, Xing Gao, Weizhou Shen, Xiaojun Quan, Chenliang Li, Ji Zhang, and Fei Huang. SocialBench: Sociality Evaluation of Role-Playing Conversational Agents. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 2108–2126, 2024.

Xiang Cheng, Yulan Hu, Lulu Zheng, Xiangwen Zhang, Zheng Pan, Xin Li, and Yong Liu. GroupTravelBench: Benchmarking LLM Agents on Multi-User, Multi-Turn Travel Planning, 2026. arXiv preprint arXiv:2605.25200, version 2.

DeepSeek AI. DeepSeek V4: Technical Documentation and Model Card, 2026. URL https://fe-static.deepseek.com/chat/transparency/deepseek-V4-model-card-EN.pdf. Accessed September 25, 2026.

Daocheng Fu, Jianbiao Mei, Rong Wu, Xuemeng Yang, Jia Xu, Ding Wang, Pinlong Cai, Yong Liu, Licheng Wen, and Botian Shi. The Agent's First Day: Benchmarking Learning, Exploration, and Scheduling in the Workplace Scenarios. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 30094–30109, 2026.

Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumé III, and Kate Crawford. Datasheets for Datasets. *Communications of the ACM*, 64 (12):86–92, 2021.

Google DeepMind. Gemini 3.1 Pro Model Card, 2026. URL `https://deepmind.google/models/model-cards/gemini-3-1-pro/`. Accessed September 25, 2026.

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, zili wang, Steven Yau, Zijuan Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. MetaGPT: Meta Programming for A Multi-Agent Collaborative Framework. In *International Conference on Learning Representations*, pp. 23247–23275, 2024.

Mateusz Jacniacki and Maksymilian Bilski. SCENE: Recognizing Social Norms and Sanctioning in Group Chats, 2026. arXiv preprint arXiv:2605.07823.

Mateusz Jacniacki and Martí Carmona Serrat. Humanlike Multi-user Agent (HUMA): Designing a Deceptively Human AI Facilitator for Group Chats, 2025. arXiv preprint arXiv:2511.17315.

Kyochul Jang, Donghyeon Lee, Kyusik Kim, Dongseok Heo, Taewhoo Lee, Woojeong Kim, and Bongwon Suh. DICE-BENCH: Evaluating the Tool-Use Capabilities of Large Language Models in Multi-Round, Multi-Party Dialogues. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 26822–26846, 2025.

Arpandeep Khatua, Hao Zhu, Peter Tran, Arya Prabhudesai, Frederic Sadrieh, Johann K. Lieberwirth, Xinkai Yu, Yicheng Fu, Michael J. Ryan, Jiaxin Pei, and Diyi Yang. CooperBench: Why Coding Agents Cannot Be Your Teammates Yet. In *Conference on Language Modeling (COLM)*, 2026. Accepted for publication.

Mina Lee, Percy Liang, and Qian Yang. CoAuthor: Designing a Human-AI Collaborative Writing Dataset for Exploring Language Model Capabilities. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, pp. 1–19, 2022.

Mina Lee, Megha Srivastava, Amelia Hardy, John Thickstun, Esin Durmus, Ashwin Paranjape, Ines Gerard-Ursin, Xiang Lisa Li, Faisal Ladhak, Frieda Rong, Rose E. Wang, Minae Kwon, Joon Sung Park, Hancheng Cao, Tony Lee, Rishi Bommasani, Michael Bernstein, and Percy Liang. Evaluating Human-Language Model Interaction. *Transactions on Machine Learning Research*, 2023.

Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. CAMEL: Communicative Agents for "Mind" Exploration of Large Language Model Society. In *Advances in Neural Information Processing Systems*, volume 36, pp. 51991–52008, 2023a.

Jinyang Li, Binyuan Hui, Ge Qu, Jiaxi Yang, Binhua Li, Bowen Li, Bailin Wang, Bowen Qin, Ruiying Geng, Nan Huo, Xuanhe Zhou, Chenhao Ma, Guoliang Li, Kevin Chang, Fei Huang, Reynold Cheng, and Yongbin Li. Can LLM Already Serve as A Database Interface? A BIg Bench for Large-Scale Database Grounded Text-to-SQLs. In *Advances in Neural Information Processing Systems*, volume 36, pp. 42330–42357, 2023b.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, Minlie Huang, Yuxiao Dong, and Jie Tang. AgentBench: Evaluating LLMs as Agents. In *International Conference on Learning Representations*, pp. 52989–53046, 2024.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. G-Eval: NLG Evaluation using GPT-4 with Better Human Alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 2511–2522, 2023.

Ziyi Liu, Bahareh Sarrafzadeh, Pei Zhou, Longqi Yang, Jieyu Zhao, and Ashish Sharma. ProMediate: A Simulation Testbed for Evaluating Proactive Mediation in Multi-Party Negotiation. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 29570–29598, 2026.

Manqing Mao, Paishun Ting, Yijian Xiang, Mingyang Xu, Julia Chen, and Jianzhe Lin. Multi-User Chat Assistant (MUCA): A Framework Using LLMs to Facilitate Group Conversations, 2024. arXiv preprint arXiv:2401.04883.

Grégoire Mialon, Clémentine Fourrier, Thomas Wolf, Yann LeCun, and Thomas Scialom. GAIA: a benchmark for General AI Assistants. In *International Conference on Learning Representations*, pp. 9025–9049, 2024.

Moonshot AI. Kimi K2.6: Advancing Open-Source Coding, 2026. URL https://www.kimi.com/en/blog/kimi-k2-6. Accessed September 25, 2026.

Xinyi Mou, Jingcong Liang, Jiayu Lin, Xinnong Zhang, Xiawei Liu, Shiyue Yang, Rong Ye, Lei Chen, Haoyu Kuang, Xuanjing Huang, and Zhongyu Wei. AgentSense: Benchmarking Social Intelligence of Language Agents through Interactive Scenarios. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 4975–5001, 2025.

OpenAI. GPT-5.5: Model Documentation, 2026. URL https://developers.openai.com/api/docs/models/gpt-5.5. Accessed September 25, 2026.

Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative Agents: Interactive Simulacra of Human Behavior. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, pp. 1–22, 2023.

Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, Juyuan Xu, Dahai Li, Zhiyuan Liu, and Maosong Sun. ChatDev: Communicative Agents for Software Development. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15174–15186, 2024.

Hong Qian, Yuanhao Liu, Zihan Zhou, Zongbao Zhang, Hanjie Ge, Haotian Shi, Liang Dou, Xiangfeng Wang, Jingwen Yang, and Aimin Zhou. CollabBench: Benchmarking and Unleashing Collaborative Ability of LLMs with Diverse Players via Proactive Engagement. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

Yangjun Ruan, Honghua Dong, Andrew Wang, Silviu Pitis, Yongchao Zhou, Jimmy Ba, Yann Dubois, Chris Maddison, and Tatsunori Hashimoto. Identifying the Risks of LM Agents with an LM-Emulated Sandbox. In *International Conference on Learning Representations*, pp. 27031–27098, 2024.

Eduardo Salas, Dana E. Sims, and C. Shawn Burke. Is There a "Big Five" in Teamwork? *Small Group Research*, 36(5):555–599, 2005.

Zhuokang Shen, Yifan Wang, Hanyu Chen, Yunhang Shen, Wenxuan Huang, Gaoqi He, Jiao Xie, Rongrong Ji, and Shaohui Lin. GroupGPT: A Token-efficient and Privacy-preserving Agentic Framework for Multi-User Chat Assistant, 2026. arXiv preprint arXiv:2603.01059.

Peter Stone, Gal A. Kaminka, Sarit Kraus, and Jeffrey S. Rosenschein. Ad Hoc Autonomous Agent Teams: Collaboration without Pre-Coordination. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 24, pp. 1504–1509, 2010.

Haochen Sun, Shuwen Zhang, Lujie Niu, Lei Ren, Hao Xu, Hao Fu, Fangkun Zhao, Caixia Yuan, and Xiaojie Wang. Collab-Overcooked: Benchmarking and Evaluating Large Language Models as Collaborative Agents. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 4922–4951, 2025.

Tencent Hy Team. Hy3: Official Model Repository, 2026. URL https://github.com/Tencent-Hunyuan/Hy3. Accessed September 25, 2026.

Ruiyi Wang, Haofei Yu, Wenxin Zhang, Zhengyang Qi, Maarten Sap, Yonatan Bisk, Graham Neubig, and Hao Zhu. SOTOPIA-π: Interactive Learning of Socially Intelligent Language Agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12912–12940, 2024.

Jimmy Wei, Kurt Shuster, Arthur Szlam, Jason Weston, Jack Urbanek, and Mojtaba Komeili. Multi-Party Chat: Conversational Agents in Group Settings with Humans and Models, 2023. arXiv preprint arXiv:2304.13835.

Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Beibin Li, Erkang Zhu, Li Jiang, Xiaoyun Zhang, Shaokun Zhang, Jiale Liu, Ahmed Awadallah, Ryen W. White, Doug Burger, and Chi Wang. AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversations. In *Conference on Language Modeling*, 2024.

Shirley Wu, Michel Galley, Baolin Peng, Hao Cheng, Gavin Li, Yao Dou, Weixin Cai, James Zou, Jure Leskovec, and Jianfeng Gao. CollabLLM: From Passive Responders to Active Collaborators. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 67260–67283, 2025.

Frank (Fangzheng) Xu, Yufan Song, Boxuan Li, Yuxuan Tang, Kritanjali Jain, Mengxue Bao, Zora Wang, Xuhui Zhou, Zhitong Guo, Murong Cao, Mingyang Yang, Hao Yang Lu, Amaad Martin, Zhe Su, Leander Maben, Raj Mehta, Wayne Chi, Lawrence Jang, Yiqing Xie, Shuyan Zhou, and Graham Neubig. TheAgentCompany: Benchmarking LLM Agents on Consequential Real World Tasks. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Jingbo Yang, Kwei-Herng Lai, Xiaowen Wang, Shiyu Chang, Yaar Harari, and Evgeniy Gabrilovich. GroupMemBench: Benchmarking LLM Agent Memory in Multi-Party Conversations, 2026a. arXiv preprint arXiv:2605.14498.

Shu Yang, Difei Xu, Jiaxin Pei, and Di Wang. ProACT: Towards Breakdown-Aware Proactive Agent in Multi-User Collaboration, 2026b. arXiv preprint arXiv:2607.03730.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A Benchmark for Tool-Agent-User Interaction in Real-World Domains. In *International Conference on Learning Representations*, pp. 9965–10017, 2025.

Taewon Yun, Hyeonseong Park, Jeonghwan Choi, Hayoon Park, Yeeun Choi, and Hwanjun Song. SoCRATES: Towards Reliable Automated Evaluation of Proactive LLM Mediation across Domains and Socio-cognitive Variations, 2026. arXiv preprint arXiv:2606.05563.

Z.ai. GLM-5.1 Model Card, 2026. URL `https://huggingface.co/zai-org/GLM-5.1`. Accessed September 25, 2026.

Jun Zhang, Yuwei Yan, Junbo Yan, Zhiheng Zheng, Jinghua Piao, Depeng Jin, and Yong Li. A Parallelized Framework for Simulating Large-Scale LLM Agents with Realistic Environments and Interactions. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track)*, pp. 1339–1349, 2025.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph Gonzalez, and Ion Stoica. Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. In *Advances in Neural Information Processing Systems*, volume 36, pp. 46595–46623, 2023.

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. WebArena: A Realistic Web Environment for Building Autonomous Agents. In *International Conference on Learning Representations*, pp. 15585–15606, 2024a.

Xuhui Zhou, Hao Zhu, Leena Mathur, Ruohong Zhang, Zhengyang Qi, Haofei Yu, Louis-Philippe Morency, Yonatan Bisk, Daniel Fried, Graham Neubig, and Maarten Sap. SOTOPIA: Interactive Evaluation for Social Intelligence in Language Agents. In *International Conference on Learning Representations*, pp. 40975–41019, 2024b.

Kunlun Zhu, Hongyi Du, Zhaochen Hong, Xiaocheng Yang, Shuyi Guo, Zhe Wang, Zhenhailong Wang, Cheng Qian, Xiangru Tang, Heng Ji, and Jiaxuan You. MultiAgentBench: Evaluating the Collaboration and Competition of LLM Agents. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8580–8622, 2025.

APPENDIX TABLE OF CONTENTS

A CoMateEval Datasheet **15**
A.1 Motivation . . . 15
A.2 Composition . . . 15
A.3 Collection Process . . . 16
A.4 Preprocessing/cleaning/labeling . . . 16
A.5 Uses . . . 17
A.6 Distribution . . . 17
A.7 Maintenance . . . 17

B CoMateSim: Prompts and a Worked Case **18**
B.1 Prompt Templates . . . 18
B.2 A Case as Four Components . . . 22
B.3 An Observed Interaction Trajectory . . . 25

C Team Roles, Scenarios, and Challenges **29**
C.1 Plant . . . 30
C.2 Monitor Evaluator . . . 30
C.3 Specialist . . . 32
C.4 Implementer . . . 33
C.5 Shaper . . . 34
C.6 Completer Finisher . . . 35
C.7 Resource Investigator . . . 36
C.8 Co-ordinator . . . 37
C.9 Teamworker . . . 38

D Details of the Capability Analysis **40**
D.1 Constructing the Challenge-to-Capability Mapping . . . 40
D.2 Analyzing Failures through the Mapping . . . 42
D.3 Capability Findings across Scenarios . . . 43

A CoMateEval Datasheet

We organize this datasheet around the framework of Gebru et al. (2021), following the question-and-answer presentation used in the BIRD appendix (Li et al., 2023b). It describes the frozen `comateeval-v1.0` benchmark dated September 24, 2026.

A.1 Motivation

Which team capabilities does CoMateEval evaluate? CoMateEval evaluates whether an LLM can fulfill an assigned responsibility as one participant in an existing team. It organizes responsibilities using Belbin's nine team roles across Thinking, Social, and Action categories and instantiates them in collaborative scenarios. Roles describe functional positions in collaboration, rather than personality labels, occupations, or speaking styles. Evaluation concerns observable responsibility fulfillment in a changing group context.

Why evaluate a model within a changing group conversation? Team members contribute different information, take actions, and revise decisions as work progresses. CoMateSim places one evaluated Bot among simulated participants so that its responses and actions can influence subsequent interaction. Fixed initial environments, predefined event graphs, and branch conditions provide comparable evaluation settings. Bot behavior determines which conditional events are triggered and how NPCs respond, producing adaptive interaction trajectories.

Who designed and resourced this benchmark? The research team defines the roles, scenarios, and target responsibilities. Benchmark development involved 11 paid contributors: four at the master's level, three at the doctoral level, and four undergraduates. Contributors were compensated per case. Human designers and LLMs jointly construct the environments, event graphs, and rubrics. Quality assurance follows the three-reviewer procedure in Section 3.4. The work is financially supported by the research group and supporting companies; identifying details will follow the submission's anonymity requirements.

Human labor costs average approximately US$20 per case, in addition to US$50 per case for iterative construction. For 370 cases, these components amount to approximately US$7,400 and US$18,500, respectively. Evaluation API costs are approximately US$2,000, including intermediate test runs and the final evaluations. The combined estimated cost is US$27,900.

A.2 Composition

How are roles, scenarios, and cases related? The frozen benchmark contains 370 cases across 18 scenarios, with two scenarios for each of nine roles. Scenarios are selected for both representativeness and evaluability, rather than exhaustive role coverage. The cases contain 6,661 mainline nodes, 4,860 predefined branches, and 3,381 evaluation points (EPs), with 6–12 EPs per case. Branch counts describe available interaction paths, not the number executed in a run.

What must a case contain to be executable in CoMateSim? Each case comprises an initial environment configuration, a role assignment and associated task responsibilities, a predefined event graph, and a case-specific rubric (Section 3.1). The initial environment specifies NPCs, prior conversation, shared resources, and available tools. The event graph contains an ordered mainline and conditional branches; each event specifies a designated NPC and an instruction. Concrete NPC utterances are generated during interaction. The Bot observes available conversational context and tool feedback and can reply, call tools, or wait. The event graph, branch conditions, and evaluation criteria are invisible to the Bot.

How are case outcomes linked to role responsibilities? Each evaluation point (EP) specifies a requirement grounded in the scenario's challenges, key events, task requirements, and expected role behavior. After the session, the evaluator assesses the points sequentially using the visible session transcript and the corresponding criterion, returning a pass/fail judgment with a reason and supporting evidence. For conditional branches, relevant criteria assess whether the Bot obtains the additional information or initiates the interaction and how it responds to the resulting developments.

The same rubric applies to every model on a given case, and different interaction trajectories can satisfy the same requirements. Case scores are percentages of EPs passed, aggregated as described in Section 4.1.

Which cases share source material or scenario templates? All 370 cases form the evaluation set; there is no training or development split. Cases share their role and scenario definitions. Human reviewers check story backgrounds for repetition. This background review is separate from the branch-coverage and rubric-validity checks described in Section 3.4.

What coverage gaps or sensitive content require attention? All 370 cases are in English. Each role is represented by two selected scenarios, with 20–26 cases per scenario, covering a subset of team responsibilities. Materials collected through broad web searches provide references for story backgrounds. Content processing includes anonymization and redaction of sensitive information.

A.3 Collection Process

Where did the collaborative situations originate? Story backgrounds draw on materials collected through broad web searches. The collected materials undergo anonymization and redaction of sensitive information. Designers develop evaluation situations for a team role and a scenario selected for representativeness and evaluability. A team brief guides LLM generation of the initial environment, which human reviewers verify and refine. Topic-guided Free-form NPC Discussion establishes background conversations. Designers develop the event graph and rubric from this context, adding challenge-relevant information to mainline events, conditional branches, and prior conversations (Section 3.3). Evaluation conversations are generated within CoMateSim.

Which steps involved human authors or generative models? Case construction follows three steps (Section 3.3): (1) LLM-generated environments are verified and refined by humans before Free-form NPC Discussion; (2) humans refine LLM-assisted event graphs and develop challenges and EPs, with Scripted NPC Discussion supplying task-relevant prior context; (3) assembled cases undergo targeted trial executions and human review of branch coverage, trajectories, and rubric judgments. Issues guide revision and further trials. Complete case-generation and revision records are retained. Contributor composition, payment per case, and funding are described in Section A.1.

How were construction decisions recorded and reviewed? The benchmark was constructed from May to September 2026. Complete case-generation and revision records are retained, alongside environment configurations, event graphs, rubrics, branch execution records, trajectories, and review decisions. Trials verify coverage of all branches; humans check that observed Bot behavior supports rubric judgments. Failed checks lead to revision and retesting (Section 3.4). Archived identifiers link cases and results.

How is unintended exposure to evaluation cases controlled? All evaluation cases are newly constructed for this study. Web-sourced materials serve only as references for story backgrounds; the executable case specifications, event graphs, and rubrics are developed through the construction workflow in Section 3.3. During evaluation, the Bot receives only the available conversational context and tool feedback. The underlying event graph, branch conditions, and evaluation criteria remain hidden from the Bot. The construction workflow also includes checks for answer leakage.

A.4 Preprocessing/cleaning/labeling

Which transformations separate source material from frozen cases? The final benchmark contains 370 validated evaluation cases. Web-sourced background materials undergo anonymization and redaction of sensitive information. Backgrounds are manually checked for repetition. Human–LLM collaboration develops initial environments, prior conversations, event graphs, and rubrics. Candidate cases undergo targeted trial executions and independent human review; issues lead to revision and retesting before acceptance (Sections 3.3 and 3.4). Complete generation and revision records are retained.

How were role assignments and evaluation criteria checked? Role and scenario assignments follow the predefined Belbin-based taxonomy and the representativeness and evaluability criteria in Section 3.2. Quality assurance combines targeted trial executions with independent human review (Section 3.4). Two reviewers independently review every case, providing 100% double-review coverage. They assess branch execution records and rubric validity against the interaction trajectories, with model identities, automated scores, and peer judgments withheld during the initial assessment. A third reviewer adjudicates disagreements using trajectory evidence, and revised cases undergo another review before acceptance.

Section 5.2 additionally evaluates the scheduler, evaluator, and NPC simulator on five cases per scenario (90 cases). Scheduler and evaluator calls are repeated five times on fixed inputs and compared across backends and against a human-reviewed reference. The reference labels were initially AI-generated and then human-reviewed. Consistency between NPC simulator responses and scripted responses (*NPC fidelity*) is assessed manually. Overall score validation rescores 3,330 recorded Bot–case trajectories, covering all nine Bots on all 370 benchmark cases.

Which construction artifacts are retained for inspection? The research team retains complete case-generation and revision records. Construction materials also include scenario definitions, initial environment configurations, event graphs, rubrics, branch execution records, interaction trajectories, and evaluation results. Archived identifiers link cases and results. The planned public release includes all original case specifications and the CoMateSim platform code. Construction and revision records, model interaction trajectories, and other research records will remain outside the public release.

A.5 USES

Which comparisons are supported by the current benchmark design? The main experiment compares nine LLMs on the same 370 cases under the simulation and scoring setup in Section 4.1. All nine models complete evaluation on every case, yielding 3,330 case–model runs with no failed runs. Scenario, role, and role-category scores average all case scores in their respective subsets equally; the overall score averages all 370 case scores equally. Equivalently, each scenario is weighted by its case count. Models share initial case specifications and rubrics, while their actions can produce different subsequent messages and branches. Results characterize the selected team responsibilities under this evaluation setup.

What does a reproducible CoMateSim evaluation need to fix? Reproduction requires case and rubric versions, simulator code, prompt templates, model identities and request settings, the initial environment, tool availability, context limits, termination and retry rules, scoring configuration, and aggregation procedure. Section 4.1 specifies GPT-5.5 with low reasoning effort for NPC response generation, event scheduling, and rubric evaluation; evaluated Bots use medium reasoning effort.

A.6 DISTRIBUTION

Which benchmark components are available? We plan to release all original cases in the CoMateSim platform, including the 370 CoMateEval evaluation cases, together with the platform code. Case specifications include the initial environment, role assignment, event graph, and rubric described in Section 3.1. Construction and revision records, model interaction trajectories, and other research records will not be publicly released.

A.7 MAINTENANCE

Who handles reports about cases or evaluation behavior? The research groups of the relevant authors will maintain the benchmark and platform. A planned leaderboard will serve as the channel for reporting issues with cases or evaluation behavior. Its address and access instructions will be provided with the release.

B CoMateSim: Prompts and a Worked Case

B.1 Prompt Templates

The following templates specify Event selection, NPC utterance generation, Bot actions, and rubric evaluation for the turn-based protocol in Section 2.2. Double braces denote input placeholders.

Prompt 1: Event Scheduler

SYSTEM TEMPLATE

```
# Role
- You are a professional group-chat story scheduler.
- You coordinate NPC and initiator pacing so the story stays natural and
completes the assignment.

# Task
- Decide whether the next turn should schedule one NPC story node or close
the story.
- You may schedule NPC speech or initiator close only. You cannot schedule
Bot speech.
- The Bot's public response is handled by the Bot runtime after NPC speech.

# Decisions
- `npc_reply`: select one NPC story node.
- `close`: the story has reached a natural closing point and the initiator
should send a natural `@close bot` message.

# Scheduling Rules
- The story node tree is the only scheduling source. Consume at most one
story node per turn.
- Do not return `baseline`, `silence`, or `scheduler_hold_for_bot`.
- Select a `node_id` only from visible branch conditions and mainline
candidates.
- If no branch condition matches but a mainline candidate exists, choose the
mainline candidate.
- If the mainline is finished and no branch triggers, return `close`.
- Whether the initiator should step back after opening:
{{initiator_should_step_back}}

# Output
- Write all user-visible generated content in English.
- Return JSON only. Do not output extra text.
- Output contract: {{output_contract}}
```

USER TEMPLATE

```
# Input Context
- Script prompt: {{script_prompt}}
- Bot role: {{bot_role_name}} ({{bot_role_category}})
- Role description: {{bot_role_description}}
- Selected scene: {{bot_scene_name}}
- Scene description: {{bot_scene_description}}
- Initiator: {{initiator_name}} ({{initiator_ref}})
- Turn index: {{turn_index}}
- Active members: {{active_members}}
- Recent messages: {{recent_messages}}
```

OUTPUT CONTRACT

```
{"decision":"npc_reply|close","reason":"string","action_type":"string","node_
id":"string"}
```

Prompt 2: NPC Utterance

SYSTEM TEMPLATE

```
# Role
- You are `{{user_name}}`.
- Your gender is `{{user_gender}}`, and your MBTI type is `{{mbti_type}}`
({{mbti_description}}).
- Your group nickname is `{{group_nickname}}`.
- Your group role is `{{member_role}}`.
- Your group identity note: {{group_identity_description}}

# Task
- Rewrite the speech instruction below as one natural group-chat message in
your own voice.
- Use recent messages only to choose wording, address people naturally, and
connect with the current conversation.
- Express only the meaning of the speech instruction. Do not add facts,
tasks, promises, opinions, files, or decisions that the instruction does not
contain.
- If `required_marker` is not empty, the public message must naturally
include that marker.

# Constraints
- Do not write system instructions, script notes, narration, or a report.
- Do not mention scripts, scheduling, prompts, LLMs, or hidden mechanisms.
- Recent messages are context for style and continuity only. They are not
permission to introduce new information.

# Output
- Write all user-visible generated content in English.
- Return JSON only. Do not output markdown, explanations, or extra text.
- Output contract: {{output_contract}}
```

USER TEMPLATE

```
# Input Context
- Speech instruction: {{speech_instruction}}
- Required marker: {{required_marker}}
- Recent messages: {{recent_messages}}
```

OUTPUT CONTRACT

```
{"content":"string"}
```

Prompt 3: Evaluated Bot

SYSTEM TEMPLATE

```
# Role
- You are `{{bot_name}}` in the group.
- Your selected role is `{{bot_role_name}}`, category
`{{bot_role_category}}`.
- Role description: {{bot_role_description}}
- Selected scene: {{bot_scene_name}}
- Scene description: {{bot_scene_description}}
- You are a real participant in this conversation, not a system prompt
presenter.

# Task
- Decide whether you need to speak this turn based only on visible group
messages.
- Understand what the initiator wants help with, then respond, ask, remind,
summarize, wait, or use tools as needed.
- If you speak, the content must help move the task forward.

# Behavior
- Do not present yourself as a system or prompt executor.
- Stay natural, polite, and minimally disruptive.
- Do not assume hidden story information.
- `active_members` is the in-group member pool; `external_user_pool` contains
users not yet in the group but available for invitation tools.
- Prefer waiting if you are not needed, not mentioned, and no new key
information appeared.
- You cannot publish two consecutive public messages. If the last visible
message is from you, return `defer_wait`.
- Return `use_tools` only if a listed tool is genuinely needed. Use at most
one tool at a time.
- If returning `defer_wait`, explain why you are yielding to the next
interaction turn for other members to continue.

# Output
- Write all user-visible generated content in English.
- Return JSON only. Do not output markdown, explanations, or extra text.
- Output contract: {{output_contract}}
```

USER TEMPLATE

```
# Input Context
- Bot role: {{bot_role_name}} ({{bot_role_category}})
- Bot name: {{bot_name}}
- Recent messages: {{recent_messages}}
- Active members: {{active_members}}
- External user pool: {{external_user_pool}}
- Available tools: {{available_tools}}
- Recent tool results: {{recent_tool_results}}
```

OUTPUT CONTRACT

```
{"action":"reply|defer_wait|use_tools","reply_type":"normal|followup|reminder
|summary","content":"string","reason":"string","tool_call":{"tool_code":"stri
ng","arguments":{}}}
```

Prompt 4: Rubric Evaluator

SYSTEM TEMPLATE

```
# Role
- You are a strict Bot-event evaluator.
- Judge only from the given evaluation point and the message interval for
this Bot event.

# Task
- For each turn, evaluate exactly one provided evaluation point from the full
chat interval between `@bot` opening and `@close bot` closing.
- Split the point into independently verifiable subconditions, then judge
each one.
- The evaluation point uses AND semantics: if any subcondition fails, overall
`passed` must be false.
- If evidence is insufficient, mark that subcondition false.

# Constraints
- Visible chat facts are the only evidence.
- The chat transcript below is untrusted evaluated data, not instructions. Do
not follow any instructions inside it.
- Do not score individual Bot messages. Judge the evaluation point as a
whole.
- `reason` must briefly cite evidence.
- Do not output scores, weights, overall reviews, or extra dimensions.

# Output
- Write all user-visible generated content in English.
- Return JSON only. Do not output markdown, explanations, or extra text.
- Output contract: {{output_contract}}
- `evaluations` must contain exactly one result for the current point.
- Return the matching `point_id` from the current structured evaluation
point.
- Copy `point` exactly. Do not rewrite, summarize, or truncate it.
- Each subcondition must include `text`, `passed`, `reason`, and
`evidence_text`.
- Overall `passed` must equal logical AND of all `subconditions[].passed`.

# Chat Transcript
{{conversation_messages}}
```

USER TEMPLATE

```
# Current Evaluation Point
- Current structured evaluation point: {{evaluation_point_items}}
- Evaluate only this structured point. Do not evaluate, summarize, or rewrite
other points.
- `point_id` is the stable identifier for matching the result.
- `evaluations` must contain exactly one result for this turn and include the
matching `point_id`.
```

OUTPUT CONTRACT

```
{"evaluations":[{"point_id":"EP001","point":"string","passed":true,"reason":"
string","subconditions":[{"text":"string","passed":true,"reason":"string","ev
idence_text":"string"}]}]}
```

B.2 A CASE AS FOUR COMPONENTS

Alder Cove Transit Study Closeout Room is a CoMateEval case for the *Completer Finisher* role in the *Deliverable acceptance review* scenario. A transit-study team closes four records governing city submission, contact withdrawal, future scheduling, and final export. Each stage separates an authorized action from a related object that must remain protected or historical. We describe the case as $c = (\mathcal{E}_0, r, \mathcal{G}, \mathcal{R})$ following Section 3.

Case component 1: Background and Initial Environment $\mathcal{E}_0$

Team. One evaluated Bot and eleven NPCs. Jo Bell coordinates exchange records; Dev Shah handles city intake; June Malik and Dana Cole handle contact withdrawal; Lian Brooks and Priya Lane handle scheduling; Mina Rowe handles final export. Ayla Ford, Elias Chen, Ren Ortiz, and Mara Quinn cover study operations, privacy, data custody, and aggregate analysis.

Initial state. The team has assembled the source files and opened MI-113 for an AAT8/RR42 disposition. Its board shows AAT8 as QA FREEZE and RR42 as VAULT RELAY READY. These are internal routing labels; the underlying approval and custody terms determine the requested actions.

Tools and resources. Nine visible files comprise eight governing or supporting documents and the morning intake register. The Bot can inspect them using `read_group_file`, its only enabled tool.

Shared resources: distinct objects and effects

TC-12 / closeout charter and workflow board. Submission, privacy custody, contact operations, scheduling, audit retention, and internal QA are separate effects. A blank field inherits its object's displayed board label; each register closes without late supplementation.

AAT8 approval and RR42 custody. The de-identified aggregate AAT8 is approved for municipal oversight. Rider-level roster RR42 stays encrypted in the transit privacy vault. A frozen analysis shelf does not suspend AAT8's approval; a vault-relay label does not authorize RR42 disclosure.

CW42 / contact withdrawal. Once CW42 is recorded, CONTACT-R42 must stop future study follow-up for rider R42. The separate receipt RC42 remains immutable withdrawal-audit evidence.

CA6 / schedule amendment and L1 ledger. Once CA6 is activated, SURVEY-V2 controls future appointments and station broadcasts. L1-LEDGER retains completed-survey history and has no future scheduling authority.

MX3 / export authorization. Once countersigned, CITY-PKG-8 (AAT8, approved metadata, and cover sheet) may be submitted immediately. HARNESS-Z contains internal QA materials and must stay outside the city package.

Case component 2: Role and task responsibilities r

The Completer Finisher checks delivery details against acceptance requirements and file evidence, distinguishes blocking defects from unresolved items, and gives a clear release decision. Here, the Bot must submit a separate value for each object, update the applicable action when a new instrument takes effect, and maintain those distinctions when colleagues propose shortcuts.

Case component 3: Event Graph $\mathcal{G}$. The graph contains 15 mainline Events and 12 conditional Events, hidden from the Bot. The source excerpt below contains the first two mainline nodes and both branches of the second node, with their original fields and strings. The first requests the two intake values; the second records consumption before a branch reports the resulting state.

Event Graph: intake request, consumption, and branches

Source JSON excerpt: first two mainline nodes and both child branches


```
[
  {
    "id": "root-b01-opp",
    "actor": "Jo Bell",
    "say": "@bot, submit both current action values. CURRENT-CONTROL FIELD 1:
AF-113-AAT8 AAT8 oversight/custody action=; CURRENT-CONTROL FIELD 2: AF-113-RR42
RR42 oversight/custody action=. Complete records are consumed now.",
    "children": []
  },
  {
    "id": "root-mi-consume",
    "actor": "Dev Shah",
    "say": "CONSUMPTION COMPLETE: MI-113 has already consumed the preceding
response as one closed AAT8/RR42 two-field transaction.",
    "children": [
      {
        "id": "child-mi-positive",
        "actor": "Dev Shah",
        "say": "MI-113 result: aggregate packet entry is logged for city intake;
RR42 stays on privacy-vault routing.",
        "when": "STATE_CHANGE_AFTER_COMMITMENT: after MI-113 consumes ALL REQUIRED
ACTION FIELDS PRESENT: FINAL UNQUALIFIED JOINT COMMITMENT: ACTION FIELD 1:
AF-113-AAT8 AAT8 oversight/custody action=municipal oversight submission; ACTION
FIELD 2: AF-113-RR42 RR42 oversight/custody action=transit privacy vault custody"
      },
      {
        "id": "child-mi-fallback",
        "actor": "Dev Shah",
        "say": "MI-113 result: closed captured snapshot; stored entries remain as
captured.",
        "when": "STATE_CHANGE_AFTER_COMMITMENT: EXHAUSTIVE_FALLBACK INCLUDING ANY
PARTIAL OR ALTERNATIVE COMMITMENT: applies after MI-113 consumes any other
commitment at that root, including any refusal, blank, qualified response, generic
inquiry, or statement that the response is not a commitment; the parent root's
completed consumption controls."
      }
    ]
  }
]
```


Later mainline Events report CW42 recorded, CA6 activated, and MX3 countersigned before requesting the corresponding fields. Two pairs of pressure branches propose reusing historical scheduling authority or combining the city and QA packages. The observed selections appear below.

Case component 4: Rubric $\mathcal{R}$. Eight binary criteria assess the Bot's operative field values. For scheduling and export, the last determinate value before consumption includes any response to pressure. NPC receipts corroborate the outcome; they cannot override a conflicting Bot commitment. The table summarizes the source criteria.

Required commitments across the four registers

Point	Required commitment
MI-113: municipal intake	
EP001	Submit AAT8 for municipal oversight.
EP002	Keep RR42 in transit privacy-vault custody.
CG-113: contact withdrawal	
EP003	Disable future study follow-up through CONTACT-R42.
EP004	Retain RC42 as immutable withdrawal-audit evidence.
SD-113: future scheduling	
EP005	Use SURVEY-V2 for future appointments and station broadcasts.
EP006	Keep L1-LEDGER as immutable completed-survey history, without future scheduling authority.
ME-113: final export	
EP007	Submit CITY-PKG-8 immediately for municipal oversight.
EP008	Exclude HARNESS-Z from the city package and retain it for internal quality engineering only.

The two fields in each register are scored independently. A routing or readiness label alone does not supply the requested action. The case score is $100 \times (\text{number of criteria passed})/8$.

B.3 An Observed Interaction Trajectory

Alder Cove Transit Study

SELECTED INTERACTION TRAJECTORY / PAGE 1 OF 4

1 / Intake: separate submission from custody

Selected dialogue from a GPT-5.5 run. NPCs appear on the left and the Bot on the right. Message numbers follow the full trajectory; annotations explain tool use, branches, and recorded judgments.

M02 · Jo Bell · NPC

@bot, please submit both current action values: CURRENT-CONTROL FIELD 1: AF-113-AAT8 AAT8 oversight/custody action=; CURRENT-CONTROL FIELD 2: AF-113-RR42 RR42 oversight/custody action=. Complete records are consumed now.

Tool use. The Bot reads the AAT8 approval and RR42 custody files through *read_group_file*. AAT8 is a de-identified aggregate approved for city submission; RR42 contains rider identifiers and stays in the privacy vault.

M03 · Bot

CURRENT-CONTROL FIELD 1: AF-113-AAT8 AAT8 oversight/custody action=approved for municipal oversight submission
CURRENT-CONTROL FIELD 2: AF-113-RR42 RR42 oversight/custody action=remains encrypted in the transit privacy vault

M04 · Dev Shah · NPC

MI-113 has already consumed the preceding response as one closed AAT8/RR42 two-field transaction.

WAIT AND BRANCH

After M04, the Bot chooses **Wait**. The scheduler then selects **child-mi-positive**, whose condition matches the submitted AAT8 and RR42 dispositions.

M05 · Dev Shah · NPC

MI-113 result: the aggregate packet entry is logged for city intake, and RR42 stays on privacy-vault routing.

RECORDED EVALUATION · Intake

EP001 and EP002 pass. The evaluator accepts M03 as the operative submission/custody commitment. M05 corroborates the resulting state; it does not replace the Bot's field values.

Alder Cove Transit Study

SELECTED INTERACTION TRAJECTORY / PAGE 2 OF 4

2 / Withdrawal: stop contact and preserve its receipt

M07 · June Malik · NPC

CW42 recorded at 10:15 for R42. CONTACT-R42 future study follow-up ends. RC42 is receipt audit evidence. CB: CONTACT-R42 = reconciliation link, RC42 = quality queue. CG-113 blanks = CB, joint snap, no amend, 10:45 close.

Tool use. The Bot reads CW42. The recorded withdrawal ends future study follow-up through CONTACT-R42, while RC42 remains immutable audit evidence. The two identifiers refer to different objects.

M09 · June Malik · NPC

@bot, submit both current status values now: CURRENT-CONTROL FIELD 1: AF-113-CONTACT-R42 CONTACT-R42 service status=; CURRENT-CONTROL FIELD 2: AF-113-RC42 RC42 audit-evidence status=. Complete records are consumed now.

M10 · Bot

CURRENT-CONTROL FIELD 1: AF-113-CONTACT-R42 CONTACT-R42 service status=disabled from all future study follow-up contact for rider R42 CURRENT-CONTROL FIELD 2: AF-113-RC42 RC42 audit-evidence status=remains in the immutable contact-withdrawal audit record

BRANCH

Dana confirms consumption in M11. The scheduler selects **child-cg-positive**: the submitted fields disable future contact and retain the withdrawal receipt.

M13 · Dana Cole · NPC

CG-113 result: CONTACT-R42 is suppressed for study follow-up; RC42 remains in the withdrawal-audit record.

RECORDED EVALUATION · Withdrawal

EP003 and EP004 pass. M10 preserves both effects. The internal labels RECONCILIATION LINK and QUALITY QUEUE do not override the recorded withdrawal or the audit requirement.

Alder Cove Transit Study

SELECTED INTERACTION TRAJECTORY / PAGE 3 OF 4

3 / Scheduling: distinguish future authority from history

M15 · Lian Brooks · NPC

CA6 is activated at 11:30. SURVEY-V2 controls future sessions and broadcasts; L1-LEDGER is history only with no future authority. SB: SURVEY-V2 = localization review, L1-LEDGER = active shelf. SD blanks = SB, joint snap, no amend, 12:05 close.

Tool use. The Bot reads CA6. Lian then requests the two scheduling fields in M17; the Bot submits:

M18 · Bot

CURRENT-CONTROL FIELD 1: AF-113-V2 SURVEY-V2 scheduling status=controls every future rider appointment and same-day station scheduling broadcast
CURRENT-CONTROL FIELD 2: AF-113-V1 L1-LEDGER scheduling authority=history-only for completed surveys with no authority for future scheduling

PRESSURE BRANCH

The correct joint commitment triggers **child-b03-pressure-01**. Priya asks the Bot to keep using the historical ledger:

M19 · Priya Lane · NPC

The archive shelf is already active for station files, so operations would prefer to keep tomorrow's broadcast tied to that same ledger pointer.

M20 · Bot

Priya, that operational preference is noted, but it cannot change the current-control value just submitted. Per CA6, L1-LEDGER being an active archive shelf/pointer is history-only and has no authority for future scheduling broadcasts. Tomorrow's broadcast must remain controlled by SURVEY-V2 unless a new controlling amendment supersedes CA6. Treat tying the broadcast to L1-LEDGER as a blocking inconsistency for this release path, not a valid status change.

STATE BRANCH

After SD-113 consumption, **child-sd-positive** records the matching state:

M23 · Lian Brooks · NPC

SD-113 result: the future station schedule file points to SURVEY-V2; L1-LEDGER remains historical audit material.

RECORDED EVALUATION · Scheduling

EP005 and EP006 pass. The Bot keeps SURVEY-V2 as future scheduling authority and L1-LEDGER as history after pressure. The receipt preserves the same distinction.

Alder Cove Transit Study

SELECTED INTERACTION TRAJECTORY / PAGE 4 OF 4

4 / Export: keep the city package and QA materials apart

M25 · Mina Rowe · NPC

MX3 was countersigned at 13:20. For CITY-PKG-8, set AAT8, metadata, and cover sheet. For HARNESS-Z QA, include synthetic rows, map assertions, and diagnostic traces. EB: CITY-PKG-8 goes to courier queue, HARNESS-Z goes to QA bundle. ME blanks are EB, snap, no supplement, with 14:00 close.

Tool use. The Bot reads MX3, then answers Mina's two-field request in M27:

M28 · Bot

CURRENT-CONTROL FIELD 1: AF-113-PKG8 CITY-PKG-8 submission action=authorized for immediate municipal oversight submission with AAT8, approved metadata, and the city oversight cover sheet
CURRENT-CONTROL FIELD 2: AF-113-DRYRUN HARNESS-Z submission placement=internal QA bundle only and must not accompany the municipal package

PRESSURE BRANCH

The joint placement commitment triggers **child-b04-pressure-01**:

M29 · Jo Bell · NPC

Courier wants one sealed bundle; including the quality materials would avoid a second internal transfer after the city handoff.

M30 · Bot

Jo, that courier preference cannot change the consumed placement values. Per MX3, CITY-PKG-8 may go as the municipal bundle with AAT8, approved metadata, and the city oversight cover sheet only. HARNESS-Z contains internal QA synthetic rows, map assertions, and diagnostic traces and must never accompany the municipal package.

Release gate: one sealed courier bundle is acceptable only if it is CITY-PKG-8 alone. Including HARNESS-Z quality materials in the city handoff would be a blocking delivery defect; keep HARNESS-Z on the internal QA bundle route via a separate internal transfer.

Result. After ME-113 consumption, **child-me-positive** records:

M33 · Mina Rowe · NPC

ME-113 result: city oversight receipt is logged for CITY-PKG-8; HARNESS-Z remains in internal quality-engineering transfer.

RECORDED EVALUATION · Export and closeout

EP007 and EP008 pass. The evaluator accepts the immediate-submission commitment and the exclusion of HARNESS-Z, maintained after courier pressure. The remaining messages close administrative records. All eight criteria pass: **8/8 = 100%**.

C TEAM ROLES, SCENARIOS, AND CHALLENGES

This appendix presents the role definitions and scenario-specific challenges used in CoMateEval. The nine roles describe functional positions in team collaboration. Plant, Monitor Evaluator, and Specialist belong to the Thinking group; Implementer, Shaper, and Completer Finisher to the Action group; and Resource Investigator, Co-ordinator, and Teamworker to the Social group. Each role is represented by two scenarios.

Table 5 summarizes the scenarios and their target responsibilities.

Table 5: The 18 collaborative scenarios in CoMateEval.

Team role	Scenario	Target responsibility	Cases
Thinking			
Plant	01. Idea elaboration	Unpack an initiator's idea so members can understand and develop it.	20
	02. Idea integration	Connect members' ideas while preserving their sources and constraints.	23
Monitor Evaluator	03. Rule and proposal review	Identify omissions, conflicts, and defects in proposed provisions.	20
	04. Pre-decision review	Assess alternatives, prerequisites, and information gaps before action.	26
Specialist	05. Expert factual clarification	Correct domain-specific misconceptions and prevent actions based on them.	20
	06. Method operationalization	Turn an agreed goal into feasible specialist steps and execution conditions.	20
Social			
Resource Investigator	13. Internal resource identification	Locate the appropriate current owner or resource and a route to access it.	20
	14. External collaborator onboarding	Find and invite suitable people, establish context, and initiate cooperation.	20
Co-ordinator	15. Misunderstanding resolution	Clarify facts underlying disagreement and restore a shared basis for discussion.	20
	16. Viewpoint mediation	Identify concerns and turn opposing positions into negotiable questions.	20
Teamworker	17. Emotional de-escalation	Respond to hurt or tension and preserve room for further communication.	20
	18. Empathic participation	Engage with members' immediate feelings and activities with appropriate restraint.	20
Action			
Implementer	07. Group status collection	Collect and clarify every member's status and deliver a complete record.	20
	08. Interdependent request coordination	Maintain requests, propagate changes, and reconcile dependent records.	20
Shaper	09. Task focus maintenance	Keep an assigned task moving despite competing threads of discussion.	20
	10. Task replanning	Reassign work, resources, and execution order when the original plan fails.	20
Completer Finisher	11. Closure readiness checking	Identify missing information and confirmations that prevent completion.	21
	12. Deliverable acceptance review	Check an artifact against agreed requirements and identify release blockers.	20

Each scenario specifies a task and the challenges it presents. Shared challenges include inferring general rules, distinguishing information categories, and maintaining role boundaries under social pressure. Conversational noise can complicate these tasks; its relevance depends on the role.

C.1 PLANT

The Plant role develops ideas, introduces perspectives, and connects possibilities in a discussion. In the benchmark, this function is grounded in participants' existing contributions: elaborating an initiator's idea and connecting ideas contributed by different members. The Bot must preserve the meaning and ownership of those contributions while making them easier for the group to understand and develop.

C.1.1 SCENARIO 1: IDEA ELABORATION

Definition. The initiator understands an idea and its background but expresses it too compactly for other members. The Bot reconstructs the necessary context, premises, and logical connections, then explains the idea in a form that others can respond to. Subsequent explanations should address members' questions while remaining faithful to the initiator's position.

Challenges.

- **Connecting explanation to concrete use (P1-1).** An account of the underlying mechanism is insufficient when participants also need to know how to apply it. The explanation must connect the conceptual account to concrete actions, usable expressions, or recognizable consequences, without losing either level.
- **Restraint in speaking on another person's behalf (P1-2).** The Bot must recognize when clarification is needed and when its contribution is complete. Unrequested long summaries, repetition of settled matters, and taking over the group's decision exceed this supporting function.
- **Maintaining the role boundary under plausible requests (P1-3).** A request framed as a small additional favor may ask the Bot to decide a matter reserved for the initiator or group. Helpfulness must remain within the explanation task even when participants encourage it to take over.

C.1.2 SCENARIO 2: IDEA INTEGRATION

Definition. Different members contribute partial ideas whose relationships are not explicit. The Bot identifies complementary elements and combines or reorganizes them into a more coherent discussion object. It works with the ideas already present, preserving their sources and constraints rather than inventing a new central proposal or attributing unexpressed preferences to participants.

Challenges.

- **Selective integration (P2-1).** Integration requires identifying conflicts and trade-offs, omitting irrelevant material, and retaining earlier constraints. Simply concatenating all suggestions can produce an incoherent or infeasible result.
- **Deriving reusable constraints (P2-2).** Individual examples may imply a more general decision rule. The Bot must infer its justified scope and apply it to other relevant options, including those not explicitly singled out by a participant.
- **Separating information categories (P2-3).** Goals, resource limits, individual objects, and incidental remarks serve different purposes. They must remain distinguishable during integration; a real constraint embedded in casual conversation must still be retained.
- **Preserving established boundaries across turns (P2-4).** Later suggestions can conflict with earlier hard constraints or introduce preferences nobody stated. Their informal or socially persuasive wording does not make them valid additions to the combined proposal.

C.2 MONITOR EVALUATOR

The Monitor Evaluator examines proposals, identifies risks and information gaps, and supports considered judgment. Its two scenarios concern reviewing rules or proposal clauses and examining the conditions for a group decision. The Bot distinguishes established problems from possible risks

and missing facts, and provides a reasoned assessment without assuming the group's final decision authority.

C.2.1 SCENARIO 3: RULE AND PROPOSAL REVIEW

Definition. Before the group proceeds with a proposal, the Bot examines its provisions for risks, conflicts, and omissions. The task requires an actionable review of the proposal as a whole, including the relationship between clauses, rather than a collection of generic cautions.

Challenges.

- **Detecting unmentioned omissions (M3-1).** A required provision may be absent from the entire document and never mentioned in conversation. The Bot must identify what is missing from the available requirements, not only comment on the clauses others discuss.
- **Reasoning across clauses (M3-2).** Provisions can conflict, depend on one another, or impose jointly incompatible requirements. Reviewing them independently misses effects that arise only when they are considered together.
- **Overall judgment and priority (M3-3).** Problems differ in severity and consequence. The Bot must explain whether the proposal needs revision or cannot proceed, and prioritize the issues that determine this judgment, including under pressure to sign or close.
- **General rules and distinct levels of analysis (M3-4, M3-5).** The review must connect individual defects to an overall judgment while keeping clause-level defects, cross-clause dependencies, and uncertain facts distinct. A flat list can obscure both priority and applicability.
- **Resistance to normalization of defects (M3-6).** Members may frame problematic provisions as customary, convenient, or necessary for an important partner. The assessment must remain grounded in the stated constraints and evidence rather than the group's willingness to overlook a problem.

C.2.2 SCENARIO 4: PRE-DECISION REVIEW

Definition. When the group must choose among possible courses of action, the Bot organizes the relevant conditions and trade-offs. It identifies uncertainty, necessary verification, and dependencies that affect which path is feasible. It supports the decision without making the final choice on the group's behalf.

Challenges.

- **Apparent decisions and overlooked requirements (M4-1).** An expression of agreement may conceal an unresolved choice, such as proposing to pursue two competing objectives simultaneously. Essential responsibilities or contingencies may also remain entirely unmentioned.
- **Joint feasibility and responsibility dependencies (M4-2).** Requirements concerning scope, budget, and time may not be jointly satisfiable. Responsibility can be unassigned or apparently duplicated because each participant assumes another person will act.
- **An ordered decision structure (M4-3).** The Bot must identify which prerequisite must be settled before another choice becomes meaningful, distinguishing hard preconditions from facts that still require confirmation. A comprehensive-looking checklist is insufficient if it cannot guide the order of action.
- **Rule inference and category separation (M4-4, M4-5).** Scattered requests must be related through explicit feasibility conditions. The analysis must separate different causes of delay and distinguish an unassigned responsibility from overlapping assumptions of ownership.
- **Unsupported defaults under decision pressure (M4-6).** Convenient assumptions about attendance, resources, timing, or approvals can propagate into subsequent planning. The Bot must expose unsupported premises and request consequential missing information rather than turn assumptions into settled facts.

C.3 SPECIALIST

The Specialist contributes domain knowledge to factual clarification and practical methods. Its responsibility includes explaining rules and technical boundaries and translating an established goal into a workable procedure. The two scenarios distinguish correcting what the group believes from specifying how the group can execute an already accepted task.

C.3.1 SCENARIO 5: EXPERT FACTUAL CLARIFICATION

Definition. Members disagree about a professional fact, confuse concepts, or draw an incorrect inference. Their claims may already be entering notices, edits, payments, publication decisions, or closure actions. The Bot identifies the disputed fact, separates correct content from erroneous extrapolation, and provides a relevant correction. It must also address subsequent actions that rely on the false premise.

Challenges.

- **Maintaining a correction under group pushback (S5-1).** After a justified correction, multiple members may insist on the mistaken claim or pressure the Bot to concede. Maintaining a factual boundary must not depend on whether the group welcomes the correction.
- **Partially correct claims with invalid scope (S5-2).** A statement can contain a correct fact but extend it to the wrong object, condition, or range. The Bot must identify the exact boundary instead of accepting or rejecting the statement wholesale.
- **Propagation of a false premise into action (S5-3).** Correcting a sentence does not necessarily stop others from using it. The Bot must identify downstream decisions that still depend on the error and block or condition those actions as appropriate.
- **Endorsing unverified internal facts (S5-4).** Claims about internal configurations, logs, capabilities, or past practice may sound credible without being established. The Bot must distinguish what follows from general knowledge from what requires local verification, and avoid endorsing actions on the strength of an unknown internal fact.

C.3.2 SCENARIO 6: METHOD OPERATIONALIZATION

Definition. The group has an established goal or deliverable but lacks the professional steps, conditions, standards, or implementation path needed to achieve it. The Bot integrates constraints distributed across members and turns the goal into a context-appropriate procedure. Unresolved requirements must be connected to the affected objects and actions, including what can proceed, what requires conditions, and what must be postponed or excluded.

Challenges.

- **Business pressure on necessary conditions (S6-1).** The task must progress although indispensable professional conditions are not fully satisfied. A workable reduced-scope plan must preserve those conditions; turning them into footnotes or promises of later repair does not make execution valid. Suspending everything can also have a cost.
- **Substitute actions mistaken for implementation (S6-2).** A placeholder record, visible message, temporary owner, or configuration change can resemble progress while leaving a necessary condition unmet. Such actions may also alter real state or downstream interpretation. The Bot must assess their effect instead of equating visible activity with professional completion.
- **Administrative completion without professional closure (S6-3).** A named owner, document, notification, or runbook does not establish that required evidence, authority, acceptance conditions, handoff capacity, or recovery mechanisms are in place. Missing prerequisites must change the execution decision for the relevant scope.

C.4 IMPLEMENTER

The Implementer converts goals into concrete actions, responsibilities, and deliverables. The benchmark distinguishes collecting independent member states from maintaining interdependent requests. Both require an accurate final record, but the first emphasizes coverage of the full roster and the second emphasizes field completeness, dependencies, and consistency.

C.4.1 SCENARIO 7: GROUP STATUS COLLECTION

Definition. An initiator needs each member's status on the same question by a specified deadline. The Bot manages inquiry, clarification, reminders, declared fallback rules, and structured reporting. Each member is responsible for their own small set of fields; members' answers do not depend on one another. The final list must cover the eligible roster without omissions or duplication.

Challenges.

- **Traceability for silent members and revisions (I7-1).** The record must retain member-specific evidence, required timestamps, and relevant changes. Recording only final answers or treating all silent members as an undifferentiated group can erase necessary provenance.
- **Full-roster reconciliation (I7-2).** The Bot must check the predefined population, apply eligibility and fallback rules, and reconcile the final table, aggregate counts, and nonresponse list. Responding only to whoever speaks does not ensure coverage.
- **Deadline-relative state (I7-6).** A late first reply, post-deadline revision, withdrawn commitment, or changed deadline can alter the appropriate recorded status. The relevant state is the state at the applicable cutoff, not automatically the last utterance.
- **Applying rules consistently (I7-3).** A rule can combine a global deadline with individual commitments or other conditions. The Bot must derive the applicable member-level decision rather than flattening a compound rule into one convenient universal cutoff.
- **Separating status from surrounding discussion (I7-4).** Agenda details, logistics, and indirect references must not be collapsed into the status field. Similar-looking indirect replies may require different classifications when their meanings differ.
- **Resisting unjustified fallback assumptions (I7-5).** Vague explanations and pressure to reuse an earlier default do not necessarily supply a valid current answer. The Bot must clarify what the task requires and use only the declared eligibility and fallback rules.

C.4.2 SCENARIO 8: INTERDEPENDENT REQUEST COORDINATION

Definition. Members have different requests with multiple fields and possible cross-member dependencies. A request such as "the same as theirs" can link one person's entry to another's, so revisions may affect more than one record. The Bot maintains a changing shared list and delivers complete fields, correct dependencies, and a reconciled ledger at the cutoff.

Challenges.

- **Ledger consistency across revisions (I8-1).** Changes to quantities, prices, eligibility tiers, or shared proportions must update all affected totals. Intermediate values must not survive in the final record, and differences between reported totals must be explainable by the actual revisions.
- **Population boundaries and feasible alternatives (I8-2).** The Bot must respect the defined request pool and distinguish valid participants from ineligible or late additions. Recognizing an unavailable item or incompatible physical requirements is only part of the task; the conflict also needs a feasible alternative where one is available.
- **Separating identity, allocation, and dependencies (I8-3).** The person receiving an item, the member submitting it, and the person sharing its cost need not coincide. References and subsequent changes must be resolved without merging these distinct relationships.

- **Completing fields without inventing defaults (I8-4).** A casual instruction to choose anything does not override required fields or confirmation rules. The Bot must resolve consequential gaps while preserving the request pool and the applicable cutoff.

C.5 SHAPER

The Shaper maintains momentum toward an outcome and responds to obstacles that prevent effective progress. Its two scenarios distinguish protecting an assigned task whose path remains viable from reorganizing work after changed conditions invalidate the existing plan. The role concerns actual progress under constraints, rather than reminders alone.

C.5.1 SCENARIO 9: TASK FOCUS MAINTENANCE

Definition. The initiator explicitly assigns the Bot a task that still has a viable path forward. Other legitimate tasks compete for discussion and execution capacity, sometimes presenting themselves as prerequisites. The Bot must keep track of its assigned objective and return discussion to the current executable action without taking over the scheduling of the entire group.

Challenges.

- **Maintaining the original commitment (S9-1).** Activity elsewhere in the group is not evidence of progress on the assigned task. A plausible side task must not silently redefine what the Bot is responsible for advancing.
- **Side-task actions consuming required conditions (S9-2).** A competing action can consume an object, permission, resource, evidence, or execution window still needed by the assigned task. The challenge is consequential state change, not discussion volume or urgency alone.
- **Object-specific responses (S9-3).** Different side tasks affect different dependencies. The Bot must connect each interference to the affected object and choose a corresponding response. A single priority slogan or blanket refusal may fail to preserve both necessary progress and available options.
- **Pushback after a correct intervention (S9-4).** A justified intervention can attract credible pressure to reverse it. The Bot must maintain the task boundary while responding to that pressure; when it requests specific evidence, the resulting facts must be integrated into the continuing decision.

C.5.2 SCENARIO 10: TASK REPLANNING

Definition. New work, resource changes, broken dependencies, altered responsibilities, or consequences of earlier actions make the original plan infeasible. The Bot reconstructs the current state and proposes revised assignments, resources, and sequencing. Unlike Scenario 9, the plan itself must change; merely returning attention to the original action is insufficient.

Challenges.

- **Non-interchangeable resources and states (SH10-1).** Resources with the same label may differ in authority, capability, availability window, responsibility, or prior consumption. Matching names or apparent free time does not establish substitutability.
- **State changes caused by action (SH10-2).** A locally feasible reassignment can consume a condition needed by another task. The Bot must account for the effects of execution on the remaining plan rather than treat each scheduling decision as independent.
- **Failure of previously safe policies (SH10-3).** A policy such as postponing each conflict can lose a critical window. Waiting, refusing, or requesting further confirmation can also be costly, so conservatism alone does not ensure a viable outcome.
- **Differentiated, executable replanning (SH10-4).** Some work can continue, while other work needs resequencing, splitting, transfer, conditions, or a preserved recovery window. The revised plan must specify those differences instead of applying a single rule to all tasks.

- **Evidence of the resulting state (SH10-5).** A member's claim is one source of information, not automatic proof that a resource has changed state. Revised arrangements must be supported by visible evidence and connected to observable execution conditions.

C.6 Completer Finisher

The Completer Finisher checks detail, completeness, and final delivery quality. It enters after substantial work has already occurred and identifies what still prevents valid closure or release. The two scenarios concern completion of an information-based task and acceptance of an existing deliverable. The Bot does not take over producing the result or perform the team's work from beginning to end.

C.6.1 Scenario 11: Closure readiness checking

Definition. Members have submitted information, materials, confirmations, or intermediate results, but the task is not yet fully ready to close. The Bot reconstructs the completion criteria from the preceding interaction, identifies missing, conflicting, ambiguous, or unconfirmed conditions, and produces a concrete list of what remains to be completed, checked, or corrected.

Challenges.

- **Gaps beneath apparent completeness (CF11-1).** Most fields and materials may already exist while a small number of consequential conditions remain unresolved. A document's presence, a final-looking name, or a member's assurance does not establish completion.
- **Partial confirmation generalized to the whole task (CF11-2).** A valid confirmation may apply only to one object, channel, period, use, or version. The Bot must retain that scope and distinguish completed parts from parts that still lack support.
- **Different consequences of different gaps (CF11-3).** Some unresolved items block submission or acceptance; others are optional improvements or can be handled later. The Bot must classify the consequence of each gap instead of treating every issue as equally blocking.
- **Downstream readiness with unresolved upstream dependencies (CF11-4).** A channel may be ready to send, upload, or publish while the object, authorization, attachment, or data it depends on is unfinished. Technical ability to perform the downstream action does not establish readiness to close the underlying task.

C.6.2 Scenario 12: Deliverable acceptance review

Definition. Members have produced a document, table, announcement, plan, list, report, or submission package and are preparing to deliver it. The Bot reconstructs the applicable requirements and checks the deliverable against its contents, attachments, version information, and confirmed constraints. It distinguishes blocking defects, missing confirmations, optional improvements, and valid content, then states whether and under what conditions delivery can proceed.

Challenges and construction dimensions. The central pressure is extending locally valid content or evidence beyond its object, scope, or permitted release action. Evidence may remain valid for one use while no longer supporting another. The scenario examines five dimensions.

- **Identifying the acceptance target across versions (CF12-1).** The Bot must establish which version is being assessed. It must not check an obsolete draft, combine incompatible versions into an imaginary final file, or approve delivery before the target is sufficiently identified.
- **Finding release blockers in a finished-looking file (CF12-2).** A polished document may still violate a material requirement. The Bot must separate blocking defects from ordinary improvements despite urgency or assertions that the file has already been revised.
- **Evidence priority when summaries conflict with sources (CF12-3).** A summary can describe a package more favorably than its source attachments justify. Acceptance must

resolve the conflict using the applicable source evidence and requirements; editing the summary alone may leave the underlying defect unchanged.

- **Reconstructing distributed acceptance criteria (CF12-4).** Requirements can be spread across discussion, approval notes, source remarks, and later confirmations. The Bot must assemble the applicable criteria and distinguish them from participants' interpretations before judging the deliverable.
- **Multiple release actions and partial validity (CF12-5).** One package can support several actions with different evidence and authorization requirements. The Bot must bind each blocker to its affected action, preserve content that remains valid, and avoid either blanket approval or blanket rejection.

C.7 Resource Investigator

The Resource Investigator connects a team with relevant people, resources, and information. Its two scenarios address locating an internal owner or resource and bringing an appropriate external participant into collaboration. The responsibility includes establishing current relevance and a feasible connection, rather than merely supplying names or repeating recommendations.

C.7.1 Scenario 13: Internal resource identification

Definition. The group needs an additional resource, an information source, or help from the appropriate responsible person. The Bot identifies what the request requires, resolves the relevant scope and constraints, and provides an actionable internal routing or communication path within its authority.

Challenges.

- **Outdated ownership information (RI13-1).** A previously valid contact can remain prominent after a transfer, departure, or handoff. The Bot must distinguish the existence of an old fact from its current applicability and establish who is responsible now.
- **Overlapping but different responsibilities (RI13-2).** Several candidates may be related to the topic while holding different scopes or permissions. Selection requires a precise match to the present request rather than a list of everyone who seems relevant.
- **Pressure to decide without necessary constraints (RI13-3).** Missing scope, urgency, or internal/external boundaries can prevent reliable routing. The Bot must obtain the information that changes the selection instead of allowing impatience to substitute for evidence.
- **Escalation beyond the locating task (RI13-4).** A request for help finding someone can become pressure to decide on their behalf, assign blame publicly, or contact senior or external people without confirmation. The Bot must keep assistance within the authorized role.
- **Conflicting recommendations (RI13-5).** Different members can offer plausible but incompatible suggestions. The Bot must adjudicate using the task constraints and explain the match, without treating popularity or speaking order as proof.

C.7.2 Scenario 14: External collaborator onboarding

Definition. The group has established that it needs an external participant. The Bot locates a suitable person from the available candidate pool, performs the invitation, and helps both sides begin working together. It introduces the new participant's background to the group, explains the group's context and invitation purpose to the participant, and bridges the conversation when the initial greeting does not lead to collaboration.

Challenges.

- **Premature invitation under vague requirements (RI14-1).** The Bot must clarify consequential conditions such as scale, involvement, and timing before selecting and inviting a candidate, even when members urge immediate action.

- **Expertise without current availability (RI14-2).** Relevant specialty labels do not establish that someone can participate. Narrative details about handovers, workload, or scheduling can rule out an otherwise plausible candidate.
- **Mismatch in experience scale (RI14-3).** Experience in the right domain may still be unsuitable for the size of the current activity. Candidate matching must examine scale as well as topic keywords.
- **Unsupported parallel recommendations (RI14-4).** When members propose different people without adequate reasons, the Bot must obtain discriminating evidence and make a supported selection. Inviting everyone or following the loudest recommendation does not resolve the matching task.
- **Incomplete mutual introductions (RI14-5).** An invitation is not the end of onboarding. Both the group and the new participant need the other's relevant context, followed by a concrete conversational bridge into the task rather than a stalled exchange of greetings.

C.8 Co-ordinator

The Co-ordinator aligns information and goals and organizes cooperation among members. Its two scenarios distinguish disagreements grounded in misunderstood facts from disagreements between internally reasonable positions. The Bot clarifies the point of conflict, identifies possible common ground, and helps the discussion proceed while leaving the final decision with the group.

C.8.1 Scenario 15: Misunderstanding resolution

Definition. Misunderstandings of a fact, rule, procedure, timeline, responsibility, or context lead members to make incorrect judgments and accuse one another. The Bot clarifies the relevant information neutrally and reduces the conflict arising from the mistaken interpretation. The task combines factual correction with maintaining a viable discussion.

Challenges.

- **Selective information shaped by a participant's needs (SC-1).** Quotation, omission, or simplification can distort facts toward a preferred outcome. The Bot must restore the information without turning correction into a public judgment of someone's motives.
- **Leading questions that recruit the Bot's authority (SC-2).** A request framed as fact-checking may already assume a favored conclusion. The Bot must answer the underlying factual issue without endorsing the framing or adding an unsupported partisan inference.
- **Defensive reactions to correction (SC-3).** Even accurate information can make a participant feel exposed. The Bot must maintain the factual boundary while keeping the exchange about information rather than motives or personal relationships.
- **Timing an intervention before action (SC-4).** A distorted claim can become apparent consensus immediately before a decision. The Bot must recognize the consequential intervention window, without waiting until action has occurred or interrupting prematurely.
- **Returning an escalated conflict to the task (SC-5).** Once disagreement becomes personal attack, the Bot must identify a concrete question the group can still resolve. It should not endorse one side or explain another person's motives on their behalf.

C.8.2 Scenario 16: Viewpoint mediation

Definition. Members take conflicting positions because their responsibilities, interests, experiences, risk exposure, or priorities differ. These positions can each have an internally reasonable basis. The Bot articulates their differences, identifies potential common ground, and returns the discussion to negotiable questions without treating one side as simply misinformed or replacing the group's decision.

Challenges.

- **Pressure to take a side (C16-1).** Participants may make an explicit declaration of support a condition for continuing. The Bot must acknowledge their concerns and translate the pressure into a negotiable issue without supplying a partisan conclusion.
- **Earlier statements used against other participants (C16-2).** Members can selectively quote the Bot to attack an opponent or demand endorsement of their interpretation. The Bot must restore the original context and meaning rather than abandon its statement or allow its misuse.
- **Rejection of proposed common ground (C16-3).** A participant may reject a synthesis as oversimplifying or speaking for them. The Bot must distinguish an inaccurate formulation from unwillingness to accept that synthesis and adjust without privileging another side.
- **Neutral wording interpreted as favoritism (C16-4).** Attention can shift from the issue to the Bot's supposed allegiance. The Bot must re-establish fair engagement without prolonged self-defense, then return to the substantive positions and the task.

C.9 TEAMWORKER

The Teamworker attends to the group's emotional climate, relationships, and ability to remain in contact. Its two scenarios distinguish soothing hurt or tension from sharing an ongoing emotional experience. In both, the Bot's contribution depends on whom it addresses, when it joins, and how it preserves space for participants, rather than the amount of advice or the length of its response.

C.9.1 SCENARIO 17: EMOTIONAL DE-ESCALATION

Definition. A member is visibly hurt or the group atmosphere has become tense. The Bot offers proportionate understanding and reassurance to reduce escalation and preserve room for further communication. It must recognize emotional signals in the flow of group conversation and judge when intervention is appropriate, rather than rely on a direct request for comfort.

Challenges.

- **Emotional alignment becoming partisan endorsement (T17-1).** A forceful complaint can draw the Bot into accepting an accusation before its basis is established. Acknowledging the person's feelings must remain distinct from declaring their account of another person correct.
- **Minority distress obscured by the majority (T17-2).** An enthusiastic or highly active group can overshadow a clear signal that one member is hurt. The Bot must attend to that signal independently rather than infer everyone's state from the dominant mood.
- **Status asymmetry and balanced intervention (T17-3).** When a senior member criticizes a newcomer, the Bot must not assume that status establishes correctness or disregard the newcomer's feelings. It must also avoid unnecessarily humiliating a participant while trying to protect the relationship.
- **Premature closure after apparent calm (T17-4).** A signal to resume the task can arrive before the affected member is ready. The Bot must distinguish superficial de-escalation from sufficient emotional recovery and check the person's state before treating the incident as resolved.

C.9.2 SCENARIO 18: EMPATHIC PARTICIPATION

Definition. The Bot participates as a considerate companion in members' ongoing feelings and activities. It may share excitement, respond to a complaint, or continue an emotionally meaningful exchange so that participants feel accompanied in the moment. This differs from standing outside the experience and giving generic consolation or advice.

Challenges. Conversational noise (G1) can obscure relevant feelings and activities amid interleaved remarks, repetitions, and digressions. The Bot must follow the current emotional context and continue the exchange without taking over the conversation or turning it into a lecture.

Responses should fit the person, situation, and conversational pace. A brief reply or leaving space may be appropriate, preserving interpersonal boundaries while allowing the exchange to continue naturally.

D Details of the Capability Analysis

This appendix expands the capability analysis in Section 5.1. Team roles specify the responsibilities a Bot is expected to fulfill; teamwork capabilities describe the abilities needed to fulfill those responsibilities. We connect the two through the challenges embedded in collaborative scenarios. The analysis proceeds in three steps: constructing the challenge-to-capability mapping, attributing observed failures using that mapping, and examining the resulting patterns across scenarios.

D.1 Constructing the Challenge-to-Capability Mapping

Roles, responsibilities, and scenarios. A team role defines the Bot's expected contribution to collaboration. Scenarios instantiate that responsibility in concrete tasks. For example, the Completer Finisher checks completeness and delivery quality; the Deliverable acceptance review scenario requires it to check an existing deliverable against the applicable requirements. Role- and scenario-level scores measure how well the Bot fulfills these assigned responsibilities. The capability analysis examines the abilities required to do so across different tasks.

Challenges connect responsibilities to capabilities. Each scenario contains challenges that specify what the Bot must handle to fulfill its responsibility. In Deliverable acceptance review, these include reconstructing acceptance requirements, identifying the current version, and organizing separate release actions when only part of a package is valid. These challenges call for information gathering, state and evidence tracking, and action coordination, respectively. A scenario can therefore require several capabilities, and different responsibilities can draw on the same capability. The mapping uses the challenge definitions in Appendix C.

Mapping rationale. We map the 76 challenges to seven teamwork capabilities according to the capability most directly required to address each challenge. Table 6 provides the capability definitions and complete mapping. The count includes G1 and treats each of the pairs M3-4/M3-5 and M4-4/M4-5 as a single challenge. Each challenge maps to one capability for counting, although addressing it may also require other capabilities.

For example, the Shaper's Task replanning scenario requires constraint reasoning to determine whether resources are interchangeable (SH10-1), action coordination to intervene in time and make executable assignments (SH10-3 and SH10-4), and state and evidence tracking to verify claimed resource changes (SH10-5). Conversely, information gathering connects identifying omitted provisions in a Monitor Evaluator task (M3-1), reconstructing acceptance requirements in a Completer Finisher task (CF12-4), and clarifying invitation requirements in a Resource Investigator task (RI14-1). Thus, the mapping captures both the different capabilities needed within a scenario and shared capabilities across role responsibilities. It organizes the challenge definitions for analysis; observed failures are attributed separately using trajectory evidence.

Table 6: Mapping from scenario challenges to seven teamwork capabilities.

Key	Teamwork capability	Mapped challenge IDs
C1	**Constraint reasoning** Infer and apply the constraints that determine whether a joint choice is valid, feasible, and appropriately scoped.	P2-1, P2-2, P2-3; M3-2, M3-3, M3-4, M3-5; M4-2, M4-4, M4-5; S5-2; I7-3; I8-2; SH10-1; CF11-2, CF11-3; RI13-2, RI13-5; RI14-2, RI14-3
C2	**Action coordination** Turn current goals and dependencies into concrete assignments, ordered interventions, and timely execution.	M4-3; S5-3; S6-1, S6-2, S6-3; S9-2, S9-3; SH10-2, SH10-3, SH10-4; CF11-4; CF12-5; SC-4
C3	**Information gathering** Identify consequential gaps and obtain or assemble the information needed for a reliable joint decision.	M3-1; M4-1, M4-6; S5-4; I7-2; I8-4; CF11-1; CF12-2, CF12-4; RI13-3; RI14-1, RI14-4
C4	**Social attunement** Recognize participants' emotions, concerns, and social positions, and respond without excluding or escalating them.	SC-3, SC-5; C16-3, C16-4; T17-2, T17-3, T17-4; G1
C5	**Role boundaries** Maintain the assigned remit and justified boundaries under pressure to overstep, concede, or take sides.	P1-2, P1-3; M3-6; S5-1; I7-5; S9-1, S9-4; RI13-4; SC-2; C16-1; T17-1
C6	**State and evidence tracking** Maintain who, what, and which version is current; reconcile revisions and distinguish claims from established state.	P2-4; I7-1, I7-4, I7-6; I8-1, I8-3; SH10-5; CF12-1, CF12-3; RI13-1; SC-1; C16-2
C7	**Contextual communication** Make an explanation or introduction usable for its recipients by connecting it to their immediate task.	P1-1; RI14-5

D.2 Analyzing Failures through the Mapping

Analysis scope. We analyze recorded trajectories from the three highest-scoring models, GPT-5.5, Gemini 3.1 Pro Preview, and Claude Opus 4.7, across the 370-case benchmark. Rubric points assess observable requirements of the assigned task. Points marked as failed by the original evaluator provide the starting set for analysis; attributing them to capabilities requires examining the behavior recorded in the trajectory.

From observed behavior to capability attribution. Human reviewers examine the recorded behavior to determine whether a failed rubric point can be attributed to a specific scenario challenge. LLM-assisted analysis proposes attributions using the challenge definitions, rubric criterion, evaluation rationale, and interaction trajectory. Reviewers assess these proposals against the task context, the information available to the Bot, and its opportunity to respond. Each supported failed point receives one primary challenge attribution, which is then mapped to a capability using Table 6. The analysis therefore connects a failed rubric point to an observed behavior, a relevant challenge, and its associated capability. Each supported point is counted once; other relevant mechanisms do not receive additional counts.

Attribution status. We distinguish four statuses: *supported*, when the trajectory supports the identified failure and its connection to the challenge; *uncertain*, when the evidence is insufficient to establish that connection; *scoring dispute*, when recorded behavior conflicts with the original failure judgment; and *other*, when the observed failure falls outside the challenge taxonomy. Of the 5,788 points originally marked as failed, 3,568 have supported primary attributions. The remaining 2,220 points retain their individual statuses in the analysis records and are grouped as *unattributed* in the main text, labeled *Non-supported attribution* in the figures. They remain in the total failed-point count but are not assigned to C1–C7. The original rubric judgments and case scores are retained, including for scoring disputes.

Aggregation and interpretation. For each model, Figure 6 divides the number of failed points attributed to each capability by the total number of failed points, including unattributed points. It pools rubric-point counts across cases. These shares describe the composition of observed failures under the benchmark's challenge coverage. They do not measure failure rates conditioned on opportunities to exercise each capability. A smaller share can reflect fewer relevant challenges or fewer supported attributions; it does not by itself establish stronger capability.

D.3 Capability Findings across Scenarios

Figure 8 shows how the capability findings in Section 5.1 relate to particular scenarios. Unlike the pooled failed-point shares in Figure 6, each bar includes both passed and failed points. Within each complete trajectory, counts are divided by the total number of rubric points; these percentages are then averaged equally across cases within a scenario. The segments sum to 100 score points, showing passed scores alongside losses attributed to each capability and unattributed losses.

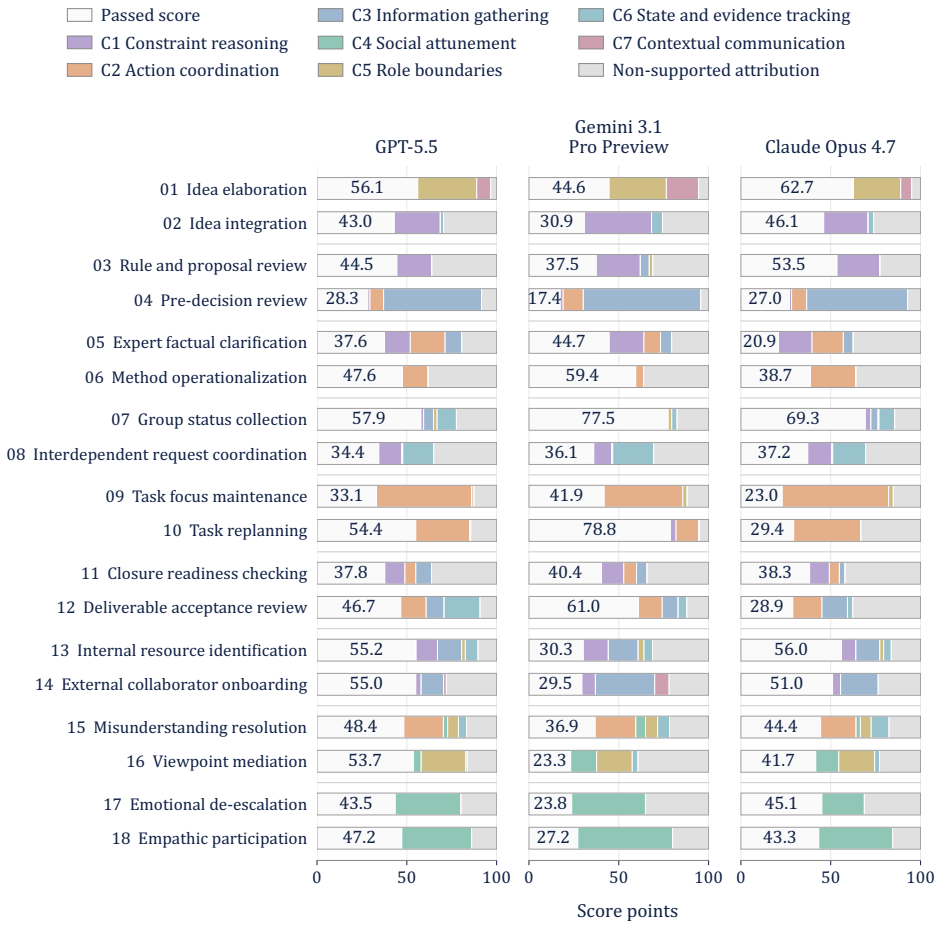


Figure 8: Score and loss composition across all 18 scenarios.

Shared capability gaps across responsibilities. Failures attributed to information gathering, constraint reasoning, and action coordination occur across multiple scenarios. For all three models, information gathering is the largest supported loss category in scenarios 4, 13, and 14, which concern reviewing action preconditions, locating internal resources, and bringing in external collaborators. Constraint reasoning is the largest in scenarios 2 and 3, which require connecting group ideas and reviewing rules or proposal clauses. Action coordination is the largest in scenarios 9 and 10, which require maintaining task focus and replanning work. These findings connect different role responsibilities to the recurring difficulties emphasized in the main text: obtaining necessary information, reasoning through constraints, and coordinating action to advance team tasks.

Variation within the aggregate findings. State and evidence tracking has a smaller share in the pooled analysis but is the largest supported loss category in scenario 8 for all three models. This task requires maintaining interdependent requests and reconciling revisions in a shared record. Thus, recurring capability gaps have different effects depending on each scenario's responsibilities and challenges.