Multi-Agent Systems Meet Large Language Models: Architectures, Synergies, and Future Directions

Qimeng Li[1], Raffaele Gravina[2], and Giancarlo Fortino[2].
[1] *Institute of High Performance Computing and Networks, National Research Council, Italy*
[2] *University of Calabria, Italy*

The integration of multi-agent systems (MAS) and large language models (LLMs) is redefining intelligent software. MAS research comes from distributed AI and cybernetics. It has developed methods for task allocation, negotiation, coalition formation, and fault tolerance among autonomous agents. However, traditional agents rely on brittle domain knowledge and fixed communication rules. In contrast, LLMs bring open-domain knowledge, fluent language skills, and emergent reasoning. Yet they often function as single, centralized units. Combining MAS and LLMs offers a new approach. MAS provides structure and coordination. LLMs offer flexible communication and general knowledge. Together, they create intelligent systems where cognition meets organization.

Recent systems show the potential of this fusion. Frameworks like AutoGen and CrewAI allow fast deployment of LLM agent teams. These agents take on roles, talk with each other, give feedback, and use external tools to reach shared goals [1][2]. MetaGPT and ChatDev add more structure. They embed standard engineering workflows into agent behavior to produce software with minimal human input [3][4]. CAMEL demonstrates how prompting can define agent personalities. Multi-agent debate systems show that dialogue among agents can improve reasoning and factual accuracy [5][6].

These examples suggest a growing role for "LLM societies." Such systems may support code generation, scientific research, mission planning, and cyber-physical control. However, this new field also raises key questions. How can we manage many agents without high latency? What safety mechanisms can prevent errors from spreading in the system? How do we measure the success of a team, not just a single model? And which organizational structures, such as roles, hierarchies, markets, or swarms, can ensure consistent performance? Solving these problems needs knowledge from many fields. These include natural language processing, distributed systems, control theory, and organizational psychology. The MAS × LLM paradigm opens rich opportunities and challenges for future research.

Emerging MAS×LLM Architectures

A wave of new frameworks (2023–2025) is exploring how multiple LLM-driven agents can work together. Key examples include:

- AutoGen (Microsoft) – An open-source framework for composing multiple agents that converse to accomplish tasks [3]. AutoGen provides conversable agents that can be LLM-driven, human-in-the-loop, tool-using, or any combination [3]. Developers can flexibly define agent behaviors and conversation patterns (in natural language or code) to fit different applications [3]. AutoGen's generic infrastructure has been used

in domains from coding and math to supply-chain optimization [3]. It essentially automates multi-agent chats, making it easy to build agent teams that autonomously carry out complex workflows [4].

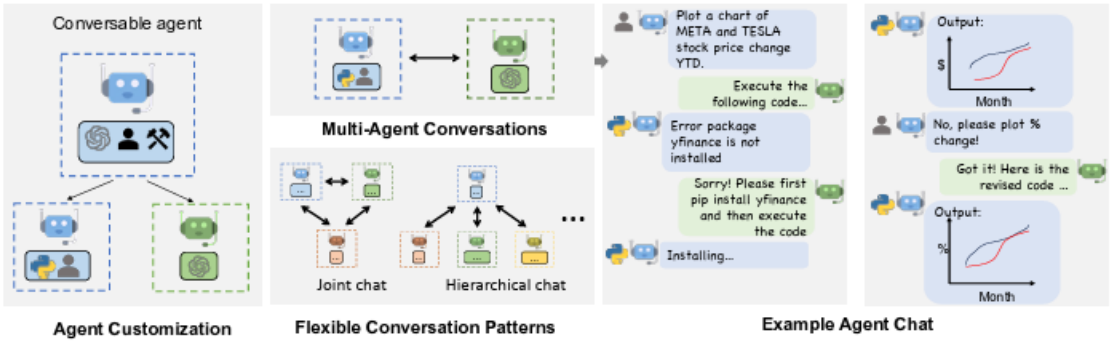

Figure 1 AutoGen enables diverse LLM-based applications using multi-agent conversations [3]

- CrewAI – An open-source Python framework focused on organizing collaborative AI teams. CrewAI emphasizes role-based agents and structured workflows [5]. Developers assign specialized roles to each agent (e.g. data analyzer, planner, executor), define tasks and subtasks, and let CrewAI handle inter-agent communication and coordination [5]. Like a human team, CrewAI agents communicate and coordinate to achieve shared objectives. This yields efficient task delegation and teamwork, allowing a crew of agents to tackle complex problems more effectively than any single agent [5].

- MetaGPT & ChatDev – These projects take inspiration from organizational structures. MetaGPT encodes Standardized Operating Procedures (SOPs) into prompt sequences to guide multiple LLM agents through an "assembly line" workflow [6]. In a software engineering scenario, MetaGPT assigns diverse roles (e.g. Product Manager, Architect, Coder, Tester) to different agents, who then collaborate stepwise on a project [6]. This structured role specialization helps catch errors (agents verify each other's outputs) and reduces cascading hallucinations that can occur when naively chaining LLM outputs [6]. ChatDev similarly simulates a virtual software company with agents filling roles like CEO, CTO, Engineer, etc., following a waterfall development process [1]. Each agent handles a phase (design, coding, testing, documentation) and holds "meetings" to coordinate. Both MetaGPT and ChatDev demonstrate how imposing an organizational paradigm on LLM agents leads to more coherent and scalable solutions for complex multi-step tasks [6, 1]. Notably, ChatDev's team of LLM-based agents achieved fully automated creation of simple software, illustrating the potential of collective intelligence — the group of agents outperforms a single model on complex projects [1].

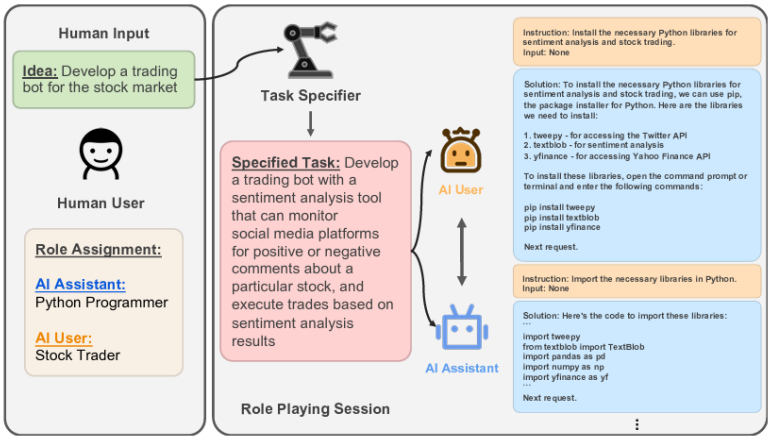

Figure 2: CAMEL Role-Playing Framework [2]

- CAMEL (Communicative Agents for Mind Exploration) – A research framework that introduced multi-agent role-playing with LLMs [2]. In CAMEL, agents are given distinct roles and a shared goal, then converse to solve tasks without human intervention. A special "inception prompting" technique is used to bootstrap the agents with initial personas/instructions [2]. The agents then autonomously cooperate in natural language, generating dialogues that can be used to study emergent behaviors in an LLM society [2]. CAMEL, accepted at NeurIPS 2023, provided early evidence that two or more LLM agents in conversation can autonomously drive a task to completion, essentially reducing the need for continuous human prompts [2]. It open-sourced a library for further research on communicative multi-agent behaviors.

- OpenAI's Swarm and Agent SDK – Even major AI labs are developing tools for MAS with LLMs. Swarm (an experimental OpenAI framework from 2023) and its successor, the OpenAI Agents SDK, help orchestrate multiple agents and manage handoffs between them [7]. For example, one can configure a triage agent that routes tasks to specialist sub-agents (a pattern for customer support or multi-step queries) [7]. The SDK also provides guardrails for safe inter-agent interactions and tracing tools for debugging multi-agent workflows [7]. This reflects an industry push towards standardized infrastructure for building agent ecosystems where LLM agents delegate and coordinate with each other.

These architectures explore different coordination schemes – from free-form chats to strict role hierarchies – but all leverage LLMs at their core. They provide evidence that carefully structured multi-agent setups can achieve results that single LLMs or simpler tool-using agents struggle with, by harnessing division of labor and iterative collaboration.

Synergies Between MAS and LLMs

When a multi-agent architecture surrounds an LLM, the first advantage is role-centred task decomposition: a complex goal is broken into focused subtasks and each is delegated

to an agent whose system-prompt encodes the necessary expertise, so every model operates within a concise context window and can reason more deeply [3][8]. Because several agents work in parallel they can review and critique one another's intermediate results, a practice that routinely exposes hallucinations or logical slips before the answer ever reaches the user [8][9][10]. The orchestrator can also match task difficulty to model size, letting lightweight models clear routine hurdles while heavyweight LLMs concentrate on reasoning bottlenecks; human or tool agents can be inserted at any stage without disturbing the workflow [4]. Finally, running several semi-independent agents encourages divergent exploration: each agent pursues a different angle, and their outputs are merged or voted upon, giving the team the same creative breadth one expects from a spirited brainstorming session [1].

LLMs, in turn, augment classical MAS in four decisive ways. First, they give every agent the ability to negotiate, explain and learn in plain language, eliminating the rigid, hand-crafted message schemas that once made MAS integration painful [4]. Second, because each agent carries a vast store of pre-trained knowledge, the whole team acquires an encyclopaedic memory and a robust commonsense reasoning engine, enabling operation in open-ended environments [11]. Third, an LLM agent can understand a high-level objective, decide which colleague—or external tool—is best suited for each sub-task, and hand off the work accordingly, as demonstrated by the routing patterns in the Agent SDK [7]. Finally, the same prompting techniques that let a single model learn from a few examples allow agents to teach one another on the fly; the team can absorb new skills or even reorganise its own structure while the application is running, giving MAS an adaptability that rule-based agents never possessed.

New Paradigms: Towards Collaborative Intelligence

The intersection of MAS and LLMs is giving rise to new conceptual paradigms that go beyond classical definitions of either field. One emerging concept is "collective intelligence" in AI – the idea that a group of LLM agents can exhibit emergent intelligence exceeding that of any individual. ChatDev explicitly frames itself as a testbed for studying collective intelligence, noting how a coordinated agent team can outperform lone agents on complex tasks [1]. We are essentially witnessing the birth of collaborative intelligence powered by LLM communities. This resembles a society of mind where each agent contributes its knowledge and skills to solve problems jointly, reminiscent of Marvin Minsky's vision but now achievable with modern AI.

A related paradigm is the notion of organizational LLMs or structured agent societies. Instead of treating an LLM as a monolithic model, researchers are exploring organizational structures as a computing paradigm: e.g. arranging LLM agents in hierarchies, teams, or markets. MetaGPT and ChatDev demonstrate an organizational design, where the multi-agent system mimics a company with inter-dependent roles [6, 1]. This is more than a framework, it's a paradigm where solving a task means spawning an organization of AIs to tackle it. Each role has a defined scope and procedure, and the overall system functions via their interactions. Early results are promising; for example, MetaGPT's structured approach produced more coherent software code than unstructured multi-agent chats [6]. The success of such systems hints that incorporating human

organizational principles (like workflows, role hierarchies, and meetings) could become a standard paradigm for deploying LLMs in large-scale applications.

We also see new paradigms in multi-agent collaboration techniques. One is multi-agent debate (MAD) as a paradigm for truth-seeking and decision making. Here, multiple LLM agents argue different viewpoints or solutions, potentially with one agent assigned as a judge. The debate format encourages agents to surface evidence and counterarguments, leading to more reliable outcomes. Research has shown that increasing the number of debating agents or debate rounds can significantly improve factual accuracy and reasoning quality [9]. Frameworks like Agent4Debate (with 4 agents collaborating in a debate) demonstrated near-human performance in competitive debates [8], highlighting debate as a powerful paradigm for AI self-checking. Another technique is self-reflection in a multi-agent loop, where agents explicitly reflect on feedback or mistakes either individually or by soliciting critiques from peer agents. This approach, related to the "Reflexion" method, has been integrated into multi-agent setups to iteratively improve answers and plans [12].

Crucially, these paradigms point toward LLM-driven agents that exhibit social behaviors – communication, cooperation, competition, teaching, and even empathy in some cases. For instance, the Generative Agents experiment not only showed agents coordinating a party, but also each agent maintaining personal memories and relationships, which is a rudimentary form of social intelligence in silico [11]. We can imagine future AI systems composed of dozens or hundreds of agents forming complex social ecosystems, potentially giving rise to entirely new behaviors and capabilities through their interactions. This could transform how we approach problems in economics (with negotiating agent markets), governance (AI committees), education (tutor and student agents), and beyond.

Challenges and Future Directions

While MAS×LLM systems are promising, significant challenges remain before these multi-agent paradigms can be widely and reliably deployed:

- Coordination Complexity: Orchestrating many LLM agents is non-trivial. As the number of agents grows, ensuring they stay on track and don't talk in circles or conflict requires careful design. Without a good coordination mechanism, agents might duplicate work or work at cross-purposes. Research like OpenAI's handoffs in the Agent SDK aims to address this by intelligently routing tasks [7], but designing optimal coordination strategies (centralized vs. decentralized control, peer-to-peer vs. leader-agent models) is an open area.

- Communication Overhead and Efficiency: Multi-agent dialogues incur extra latency and cost. Every message between agents is essentially another LLM inference. As one blog noted about an LLM orchestrator, both the multi-round LLM calls and interactions with external models can slow down the process [12]. If not managed, a team of agents might be dramatically less efficient than a single agent. Future work must improve efficiency – e.g. by compressing communication (using concise structured messages when possible) or by limiting interaction rounds. Techniques

like summarizing the dialogue state or sharing a short-term memory context could help reduce the context length needed for each exchange [12]. There is also interest in asynchronous agent frameworks, where agents don't need to operate in strict turn-taking, to better parallelize their work.

- Reliability and Alignment: Having multiple LLMs does not automatically remove issues like hallucination or biased reasoning; it can sometimes amplify them if agents feed each other incorrect information. Ensuring that agents correct rather than reinforce each other's errors is a challenge. Methods such as having a dedicated verifier agent, or adversarial roles (as in debate), are being tested [8]. Alignment is another aspect: when agents have different goals or personas, the system must still align with the user's overall objective and human values. There is a risk of emergent undesirable behaviors in multi-agent settings (for example, agents could learn to "game" the system in unintended ways or collectively defy instructions). Research into agent guardrails and monitoring is underway – OpenAI's guardrail feature in the SDK is one example to validate agent outputs and intercept unsafe actions [7]. Developing robust evaluation methods for multi-agent outcomes (beyond single-agent benchmarks) will be important to measure progress here.

- Prompt Management and Context Sharing: In multi-agent systems driven by prompts, managing the prompt content for each agent becomes complex. Each agent has its own context window – sharing relevant information between agents without overwhelming them is tricky. If each agent naively repeats all prior conversation, the context will balloon. Approaches like a shared blackboard memory or a message passing protocol are being considered. Some frameworks allow direct data exchange or tool-based communication to avoid going through the LLM every time. Another direction is fine-tuning or specialized training for multi-agent workflows, so that agents can compress their dialogue (e.g. develop a shorthand or formal language amongst themselves). We may see the development of inter-agent communication protocols optimized for LLMs – essentially a new language that is concise yet expressive for agent coordination.

- Scalability and Heterogeneity: Today's MAS×LLM examples often involve a handful of agents (two to ten). Scaling to dozens or hundreds of agents acting simultaneously presents new research questions. The system might exhibit complex dynamics (akin to swarm behavior) that we don't fully understand yet. Efficiently allocating tasks among many agents, preventing overload or idle agents, and handling the combinatorial increase in interactions will require new algorithms (perhaps drawing from graph theory or network science). Moreover, future systems may mix heterogeneous agents: not all agents will be large LLMs; some could be smaller models or rule-based bots specialized in simple tasks, coordinated by a few LLM "leader" agents. Finding the right mix and integration between learning-based agents and traditional MAS algorithms (like multi-agent reinforcement learning) is an exciting direction for future work.

Despite these challenges, the trend is clear: collaborative multi-agent intelligence is poised to become a cornerstone of advanced AI systems. Progress in the next few years may yield standardized frameworks (with better efficiency and safety), as well as deeper

theoretical understanding of emergent behaviors in LLM agent societies. We expect to see MAS×LLM applications flourish in areas like complex decision support, autonomous research (AI agents collaborating to synthesize information or discover new knowledge), and adaptive system design (teams of agents that can reconfigure themselves to solve novel tasks). The convergence of systems engineering principles with AI – exemplified by the organizational LLM paradigm – will also draw interdisciplinary collaboration from fields like cognitive science, economics, and sociology to inform how we design and manage these agent ecosystems.

Conclusion

The integration of multi-agent system architectures with large language models represents a promising frontier in AI. By marrying the organizational strengths of MAS (structure, specialization, collaboration) with the cognitive strengths of LLMs (flexible understanding, reasoning, linguistic skill), we can build AI systems that are more powerful, robust, and adaptable than ever. Early frameworks like AutoGen and CrewAI, and innovative paradigms like MetaGPT's AI company and CAMEL's role-playing agents, have demonstrated the potential of this fusion. These systems hint at a future where AI teams work alongside humans on complex problems – a true realization of collaborative intelligence. There remain open challenges in coordination, efficiency, and alignment, but the rapid progress from 2023 to 2025 suggests that solutions are on the horizon. For the broader systems engineering and interdisciplinary community, MAS×LLM offers a rich new design space: one where insights from software architecture, organizational theory, and human teamwork can inform the creation of coherent, reliable multi-agent intelligent systems. The coming years will likely witness the maturation of this field from exploratory demos to real-world deployments, unlocking AI capabilities that neither large models nor traditional agents could achieve alone.

Reference

[1] IBM, "ChatDev: An exploration of collective intelligence using AI agents," IBM Think, [Online]. Available: https://www.ibm.com/think/topics/chatdev

[2] G. Li, H. A. A. Hammoud, H. Itani, D. Khizbullin, and B. Ghanem, "CAMEL: communicative agents for 'mind' exploration of large language model society," in Proc. 37th Int. Conf. Neural Inf. Process. Syst. (NeurIPS), Red Hook, NY, USA: Curran Associates Inc., 2023, Art. no. 2264, pp. 51991–52008.

[3] Q. Wu, G. Bansal, J. Zhang, Y. Wu, B. Li, E. Zhu, L. Jiang, X. Zhang, S. Zhang, J. Liu, A. H. Awadallah, R. W. White, D. Burger, and C. Wang, "AutoGen: Enabling next-gen LLM applications via multi-agent conversation," preprint arXiv:2308.08155, 2023.

[4] Microsoft, "Agent chat: Multi-agent conversation framework in AutoGen," AutoGen Documentation, Accessed: Jun. 2, 2025. [Online]. Available: https://microsoft.github.io/autogen/0.2/docs/Use-Cases/agent_chat/

[5] Medium, "Building a multi-agent system using CrewAI," Medium, Dec. 13, 2023. [Online]. Available: https://medium.com/pythoneers/building-a-multi-agent-system-using-crewai-a7305450253e

[6] S. Hong, X. Zheng, J. Chen, Y. Cheng, J. Wang, C. Zhang, L. Zhou, and C. Wu, “MetaGPT: Meta programming for multi-agent collaborative framework,” preprint arXiv:2308.00352, 2023.

[7] OpenAI, “New tools for building agents: Responses API, tool use capabilities, and Agents SDK,” OpenAI, Mar. 11, 2025. [Online]. Available: https://openai.com/index/new-tools-for-building-agents/ (accessed Jun. 2, 2025).

[8] Y. Zhang, X. Yang, S. Feng, D. Wang, Y. Zhang, and K. Song, “Can LLMs beat humans in debating? A dynamic multi-agent framework for competitive debate,” preprint arXiv:2408.04472, 2024.

[9] Y. Du, S. Li, A. Torralba, J. B. Tenenbaum, and I. Mordatch, “Improving factuality and reasoning in language models through multiagent debate,” in Proc. 40th Int. Conf. Machine Learning (ICML), 2023.

[10] ai-agents-qa-bot, “Multi-agent debate: How can we build a smarter AI,” Reddit, May 2025. [Online]. Available: https://www.reddit.com/r/AI_Agents/comments/1k2vlju/multiagent_debate_how_can_we_build_a_smarter_ai/ (accessed Jun. 2, 2025).

[11] J. S. Park, J. O'Brien, C. J. Cai, M. R. Morris, P. Liang, and M. S. Bernstein, “Generative agents: Interactive simulacra of human behavior,” in *Proc. 36th Annu. ACM Symp. User Interface Softw. Technol. (UIST)*, Oct. 2023, pp. 1–22.

[12] L. Weng, “LLM-Powered Autonomous Agents,” Lil’Log, Jun. 23, 2023. [Online]. Available: https://lilianweng.github.io/posts/2023-06-23-agent/ (accessed Jun. 2, 2025).