International Journal of Science, Strategic Management and Technology
Volume **02** Issue **03** March-**2026** | ISSN: **3108-1762 (Online)** | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases

Beyond Large Language Models: How Small, Specialized Agents are Replacing Monolithic Systems

Pritha Nandi[1], Subhajit Manna[2]
[1,2]Department of Computer Science & Engineering Institute of Engineering & Management, Kolkata, India

https://doi.org/10.55041/ijsmt.v2i3.380

Cite this Article: Nandi, P. & Manna, S. (2026). Beyond Large Language Models: How Small, Specialized Agents are Replacing Monolithic Systems. International Journal of Science, Strategic Management and Technology, *02*(03). https://doi.org/10.55041/ijsmt.v2i3.380

License: This article is published under the Creative Commons Attribution 4.0 International License (CC BY 4.0), permitting use, distribution, and reproduction in any medium, provided the original author(s) and source are properly credited.

Abstract. For years, the AI field kept chasing the bigger models where more parameters, more compute and more cost was observed. But by 2024, teams actually deploying AI in hospitals, factories, and satellite systems started hitting a wall where large models were too slow, too expensive, and too risky for private data. This paper takes a look at the changes that have happened, since then. Numerous researchers have splitted AI into smaller, focused parts that work together. Here, instead of one expert doing everything, the focus is on a team where each person handles what they do best. It was observed that three things matter most. First, separating planning, memory, and action into distinct modules makes systems far easier to fix and upgrade. Second, mixing small models for fast edge tasks with larger models for complex reasoning cuts costs by 40-70% without loss of output quality. Third, giving agents structured ways to communicate and disagree actually improves results. Controlled disagreement turns out to be useful. It was noted that two coordination patterns gain real traction: Mixture-of-Agents, where multiple models vote on the best answer, and Communicative Agents for "Mind" Exploration of Large Language Model Society (CAMEL) [1], where agents negotiate through role-based dialogue. Both patterns appeared to be promising .But neither is production-ready without human oversight. Thus, it can be said that there is no agreed standard for how agents across different systems should talk to each other. Until that exists, reproducing results from one lab to another stays harder than it should be. This is the most important unsolved problem in this area right now.

Keywords: *Agentic AI [2], Small Language Models[3], Modular Architectures, Multi-Agent Systems [2], Edge Intelligence*

1 INTRODUCTION

A big change in the LLM-sphere was seen around 2023. AI tools stopped being just question-answering machines and they started taking real actions such as running code, browsing the web, chaining together dozens of steps to complete a task without being told what to do at each point along the way. The trouble is that most of this new agentic AI still runs on massive general-purpose models that are dependent on cloud. That works well for demo. But real deployments are much different. A hospital cannot send patient records to an external server every time a nurse needs a quick summary because the privacy risk can be a huge threat. A factory floor sensor cannot wait two seconds for a cloud response when a machine is about to fail. A satellite ground station running thousands of queries a day cannot afford ten dollars for a call. The same three complaints was seen where it was too slow, too expensive, and too risky for private data.
The term *LLM Tax* was started informally. It captured a real problem. Users pay it every time they route a straightforward

© Author(s). This work is peer-reviewed, openly published, and permanently archived
This article is openly accessible and reusable with proper attribution.

classification task through a 70-billion-parameter general model, when a 1-billion-parameter specialist would have done the job better, faster, and at a fraction of the cost. It was also observed that replacing large models with smaller task-specific ones cut operational costs by over 50 percent in real enterprise settings, without any meaningful drop in output quality. And this same result showed up across multiple independent studies. However, the core idea is not very complicated: a small model at the edge handles fast routine tasks locally, a larger model in the cloud handles only the genuinely difficult reasoning that needs it, and structured protocols tell these agents how to divide work, communicate, and disagree without producing contradictory outputs that confuse the user or break the system.

This aligns well with how Yoav Shoham described agents with separate mental roles back in 1993 [4].The new part is that now humans have the hardware, the training methods, and the deployment evidence to actually build these systems at scale and the real-world failures of large single-model deployments also motivate humans to do so.

This survey is built around three Research Questions:

- *RQ1:* How does splitting one large model into many small coordinated ones change what the architecture actually needs to do?
- *RQ2:* Do specialized small models genuinely fix the cost and latency problems, and if so, under what conditions?
- *RQ3:* What communication rules do agents need to coordinate reliably without increasing each other's mistakes, compared to what one model alone would have produced?

1.1 Contributions

The primary goal of this survey is to provide a roadmap for the post-monolithic times. The important contributions of this work are:

1. Understanding about the *Taxonomical Framework*. It helped to understand what the agent is made up of. The modular agents was split into four jobs: perceiving input, planning, storing memory and taking actions. Keeping these separate helps us to fix and replace all the parts.
2. Understanding about *Architecture* of the models, and understanding about how it actually works**.** The real performance data was checked from 2023 to 2025 for small and large model setups in hospitals and factories. It was important to not repeat what the papers claim, but to understand about architecture thoroughly.
3. Understanding what is still missing and about the future roadmap. The future roadmap helps to understand the challenges in safety that creates the roadmap for futuristic intelligence systems that will come after this. It helps to understand what is still missing. No standards exist as such for agent communication. Safety testing is weak. These gaps matter.

Thus the future of AI is not found in bigger models, but is found in smarter, more specialized systems.

2 HOW AGENTIC AI ACTUALLY DEVELOPED

Agentic AI did not appear suddenly from nowhere. Researchers had the theory decades ago but lacked the tools. The blueprint for how an AI agent should think and act became clear, when systems were observed starting from symbolic logic in 1970s to the belief-desire-intention agents. Large language models just made it more buildable.

2.1 Where the Idea Actually Started: Agent-Oriented Programming (AOP)

Back in 1993, Yoav Shoham [4] asked a question most programmers were not asking at that time. What if instead of telling a program what to do step by step, humans can assign beliefs, goals, and the ability to decide for itself to it? He called this *Agent-Oriented Programming***.** In that case the program would no longer be a list of commands but become an entity with a mental state. AOP and BDI-Belief-Desire-Intention frameworks were solid ideas that were left unused for decades [5, 6]. Theory was not the main problem. It was that the tools available such as rule engines, symbolic logic, decision trees could not handle the ambiguity of real tasks. Large language models changed that by giving agents something closer to genuine reasoning, rather than just pattern matching happening against a fixed set of rules.

© Author(s). This work is peer-reviewed, openly published, and permanently archived
This article is openly accessible and reusable with proper attribution.

2.2 Shift in Recent Times from Prompting to Iterative Planning

From 2020 to 2023, LLMs were capable of answering questions. But ReAct (Reason+Act) changed this and model could now think before acting and is also capable of trying something, seeing what happened, adjusting. Thus a single prompt can now trigger a whole chain of tasks running end to end without anyone stepping in [7].

2.3 LLM vs. Agentic Systems – What Is the Difference?

People mix these two terms constantly all the time, and it causes real confusion [8, 9].
• *Large Language Model (LLM):* It takes input, gives output, and remembers nothing between the calls as it has no tools, no memory and no goals.
• *Agentic System:* This refers to a framework that wraps the LLM within a *Stateful Environment*. This includes external memory layers which are vector databases, sensory inputs which is multimodal perception and an Action Space for API tools.
LLM does not run the agent. Instead, the agent runs the LLM, using it for reasoning .At the same time, the other parts handle memory, tools, routing, and everything else that makes the system actually work.

2.4 Small Language Models and their Specific Jobs

The LLM sits inside the agent. It does not function as an independent entity [3,10]. Irugalbandara et al. [10] and Williams [3] both found the same thing that small models trained for specific jobs outperform singular large models that do everything. There can be a system as shown in [11, 12], where one SLM handles memory, another runs code, and a third routes tasks. Bigger models does not mean better, such as in [13]. High quality Agentic AI is defined by an orchestrated network of SLMs.

3 HOW A MODULAR AGENT IS PUT TOGETHER

Breaking an agent into separate parts sounds complicated. But the authors of [14] simplifies it and used three components: *Perception, the Brain, and Actions*

3.1 Functional Component Breakdown

A three-tier structure consisting of Perception, the Brain (Cognitive Layer), and the Action Space was proposed.

1. *Perception:* By this, researchers refer to what the agent sees. Inputs are not always just text. In remote sensing, inputs are satellite images, sensor streams, and LiDAR data that need processing before any reasoning can happen, as seen in [14], by Alipour Talemi et al. They showed that keeping perception as a separate module solves this problem. A vision encoder handles the raw data and passes clean tokens to the reasoning layer, and thus the core model never has to deal with image files directly.
2. *The Brain:* The cognitive layers has two jobs - *planning and memory*.
Planning breaks the main goal into a step-by-step task list. *Memory* splits the main goal into short-term context and long-term storage via Retrieval-Augmented Generation (RAG) [15,16]. This stops the model hitting context limits on long jobs and keeps useful knowledge available between sessions.
3. *Actions:* By this humans try to understand what the agent does. It was observed that the action layer runs code, queries databases, and calls APIs using frameworks like Toolformer and Gorilla [17]. It sits apart from the brain, to perform active integration.

3.2 The Logic of Decoupling

Decoupling happens when a tightly coupled, monolithic system is broken down into independent components. .This is important because –
• *Security* is necessary. The action layer enforces permissions, thus a bad output from the cognitive layer cannot directly cause harm. All activities like file deletions, irreversible API calls, hardware commands, all will get checked first.
• Also necessary for *upgrades*. This is because swapping one small specialist model is far cheaper than retraining a

© Author(s). This work is peer-reviewed, openly published, and permanently archived
This article is openly accessible and reusable with proper attribution.

175B-parameter system from scratch.
This separation is what enables heterogenous orchestration – which simply means a team of focused parts can easily beat one large generalist model.

4 SMALL MODELS & TIERED SYSTEMS: THE PRACTICAL CASE

The real bottlenecks in agentic AI are not about model intelligence. Instead are speed, cost, privacy, and whether the system can actually run where it needs to run. Using one large model for every task regardless of how simple the task is, was demonstrated as wastage of money and fails in environments like hospitals and factory floors, where we see that data cannot leave the building.

4.1 The Rise of Small Language Models (SLMs) [10]

Small Language Models [10] typically have 1B to 7B parameters. They do well when they are trained for one specific job, rather than everything at one time. TinyLlama [18] showed a 1.1B-parameter model, and they are capable of precise reasoning on constrained tasks when they are carefully trained. MobileLLM [19] covered the same topic about devices which have tight memory limits and where intent detection, task routing, tool calls, all are running locally. This was extended into BitNet [20], which showed that 1-bit quantized models, demonstrating that representational sparsity or rarity does not prevent competence of reasoning, when the model roles are carefully scoped. Same pattern across all three models was seen , hat task fit matters more than raw size [12, 21]. Williams et al. [3] and Reynolds and Thornton [22] both arrived at the same conclusion independently that a small model doing one thing well consistently beats a large model that is trying to do everything.

4.2 Two-Tier Deployment Architecture

A *tiered deployment model* is used for understanding raw data and for reasoning.
Tier 1: Edge-Level SLMs
Tier-1 agents are deployed at the *edge*, close to the user or data source. These agents prioritize:
- Low latency
- Data locality and privacy
- High availability under limited compute

Typical tasks include:
- Intent classification
- Input sanitization and filtering of inputs
- Simple reasoning and policy checks
- Cached response generation

SLMs such as Tiny Llama or MobileLLM are good examples of this, aligned with findings from *Scaling down to scale up: A cost-benefit analysis of replacing OpenAI's LLM with open source SLMs in production*, as cited by Irugalbandara et al. [10]. This study showed that *over 60% of the enterprise agent queries can be resolved locally* without involving cloud-scale LLMs.
Tier 2: Cloud-Level LLMs
Tier-2 systems consist of *heavyweight LLMs*. They come into action when tasks exceed the reasoning or planning capacity of Edge-Level SLMs. These models handle:
- Multi-step reasoning
- Ambiguity resolution
- Binding decision in cross-agent conflict
- Long-horizon planning

This separation aligns with the *cost-benefit analysis* reported in *Scaling Down to Scale Up* [10], which showed that hybrid SLM–LLM models can reduce operational costs by *40–70%* while maintaining good output quality. And this was a finding further supported by edge-cloud collaborative frameworks [23, 24].

© Author(s). This work is peer-reviewed, openly published, and permanently archived
This article is openly accessible and reusable with proper attribution.

4.3 Logic Flow: The Router Pattern

Architectural Router: It is a lightweight module that assigns tasks to the computational path that it finds appropriate. So rather than relying on a monolithic system, the router acts as:

- A small classifier
- A SLM augmented by rules
- A gating function that is confidence-aware

This pattern aligns with orchestration principles seen in Varney's *Orchestrated Networks of Specialized Small Language Models*[25], here the *first-class agent is* Intelligent routing (or skills-based routing), consistent with broader frameworks identified in the literature [14,26], as demonstrated in Fig.1 ., as seen in following page.

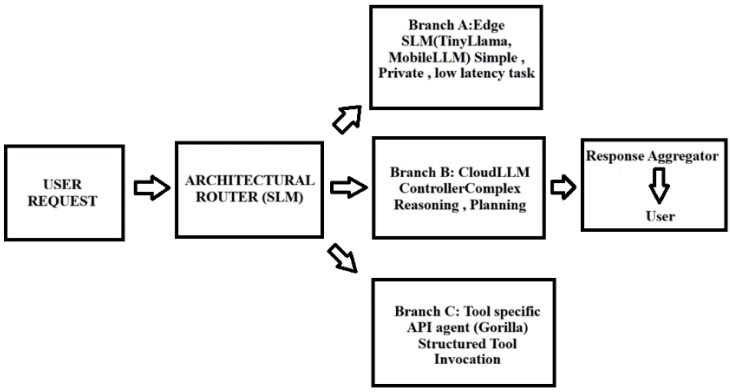


Fig. 1 *Heterogeneous routing logic for balancing latency, privacy, and reasoning depth in modular agentic systems.*

5 MULTI-AGENT COLLABORATION SYSTEM

Heterogeneous routing helped to determine *where* intelligence is executed within a modular agentic system, and a system of *multi-agent orchestration* is used. Thus rather than relying on a single reasoning structure, agentic architectures with *distributed cognition* are used, where multiple specialized agents collaborate and refine outputs iteratively. Also, running multiple agents is only useful if they can communicate, divide work, and handle disagreements without falling apart. But there might be coordination problem due to this.

5.1 Communication Protocols and Agent Handshaking

The foundational principles of **agent-oriented programming** was given by Shoham (1993) [4], which shows that agents are autonomous entities defined by beliefs, desires, and intentions. *High-quality communication protocols* between the agents is used to prevent redundancy, or premature consensus or agreement. The Communicative Agents for "Mind" Exploration of Large Language Model Society (*CAMEL)* [1] *framework* as cited by Li et al., 2023[1] introduced a formalized *agent handshaking* mechanism in which agents (i) assume predefined roles, (ii) exchange structured messages, and (iii) operate under constrained policies. More recent surveys, including *A Review of Autonomous and Collaborative Agentic AI Systems for Enterprise Applications* (2025)[2], showed that explicit communication schemas consistently beat informal prompt chaining, especially in environments that are critical for safety.

5.2 The Mixture-of-Agents (MoA) Archetype

MoA (Mixture-of-Agents) applies an all -rounded approach of learning at the agent level .Here , instead of making an average of the token probabilities, a aggregate of the full reasoning outputs was made from the multiple agents, and then the best was picked. The *standard setup* was observed which consists of a *Planner* that breaks the task down, *Executors* generate candidate answers, and a *Critic* reviews and pushes back. Reynolds and Thornton [22] emphasizes robustness through feedback loops, rather than use of mainstream centralized reasoning. LLM-Blender [27] confirmed that

© Author(s). This work is peer-reviewed, openly published, and permanently archived
This article is openly accessible and reusable with proper attribution.

ensembling smaller specialist agents can beat GPT-4 on complex benchmarks of reasoning [28, 29].Thus, we can say that MoA configuration is valuable.

5.3 Conflict Resolution and Handling of Disagreement in Multi-Agent Societies

Producing different answers is a useful signal. It is not considered to be a bug. Three resolution strategies come up consistently in the literature (1) Critic-based arbitration – where a dedicated agent reviews competing outputs and picks or synthesizes the best one, (2) Confidence – weighted voting – where the agents with stronger track records on similar tasks get more weight in the final decision, (3) Hierarchical override is also an important strategy .So, a higher-tier agent steps in only when, the other agents genuinely cannot agree. Talemi et al. [14] found that coordination that is conflict-aware, significantly improves reliability under noisy inputs. The same is seen in healthcare and enterprise settings, where agents disagree productively, and that allows to stop one wrong answer from cascading through the system [30, 31]. It is seen that all the different agents are connected with the shared memory vector database, which is a specialized storage system, designed to store and search vector embeddings, which are numerical representations of unstructured data like text, images and audio, as demonstrated in Fig. 2.

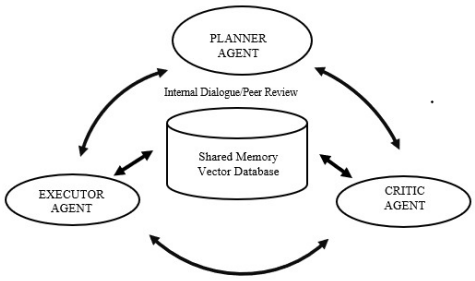


Fig. 2 *Modular multi-agent system with shared memory, role specialization, and peer-review feedback loops*

6 THE ALGORITHMIC FRAMEWORKS BEHIND AGENTIC AI

Six different algorithmic approaches are used and they have shaped how modular agentic systems are built today. Each one solves a different part of the coordination problem.

6.1 Task Planning and Tool Use

ReAct (Reason + Act Model) [7] showed that a model can think before acting, and it gives time for generating output, that helps the agents behave in a reliable way. Task Planning and Tool Usage (TPTU) [32] was built on the ReAct model, where the agents used to break a complex goal into sub-tasks and users have to pick the right tools for each one. Toolformer [33] demonstrated this same principle, where mixing reasoning and action steps also helped to produce better results. So instead of treating everything separately, this method is used.

6.2 Knowledge Distillation

Large models have a lot of knowledge. The question is how the users can get that knowledge into a small model efficiently. Orca model [34] showed this by exhibiting that GPT-4 *reasoning traces* are end-to-end records of a single request, and can be distilled into compact models. Also, Wu et al. [24] applied the same idea to the edge and mobile deployments. Gunasekar et al. [35] showed that synthetic training data can significantly reduce the gap between the small and large models' reasoning abilities.

6.3 Edge–Cloud Collaborative Inference

Cloud is not needed in all cases. Liu et al. [23] and Hao et al. [19] also showed that, splitting work across the local SLMs and cloud LLMs helped to reduce latency and cost, in a substantial amount. Also, Irugalbandara et al. [10] and Duffy et al. [36] also, confirmed these gains in real deployments.

© Author(s). This work is peer-reviewed, openly published, and permanently archived
This article is openly accessible and reusable with proper attribution.

6.4 Mixture-of-Agents

Instead of relying on one model producing one answer, MoA(Mixture-of-Agents) are used. A system collects outputs from multiple agents and picks the best. LLM-Blender [27] , showed that an ensemble of heterogenous agents beat the single-model threshold on the reasoning tasks. Wang et al. [28] also later integrated this framework.

6.5 Retrieval-Augmented Systems

The agents work better when they are able to pull in some external knowledge, rather than completely relying on what they were trained on. Edge-assisted and Collaborative Retrieval Augmented Generation (EACO-RAG) [15] combined this edge-assisted retrieval with shared knowledge updates, that was seen across the agents. Xu et al. [16] also showed that coordinated retrieval across the agents improve complex type of question -answering. This is better than each agent retrieving, in an independent manner.

6.6 Communication Protocols

Shoham [4] and Varney [25] both have put the argument that structured communication rules are not optional, they are extremely important. They helped to make multi-agent systems predictable and safe. Communicative Agents for "Mind" Exploration of Large Language Model Society (CAMEL) [1] showed role-conditioned communication, among the agents.

6.7 Summary

These six frameworks together showed that building good agentic systems can be a modelling problem, and it can also be a coordination problem. Both Williams et al. [3] and Reynolds and Thornton [22] showed that how well the agents coordinate. It is of higher importance than the performance of big individual models.

7 DOMAIN-SPECIFIC IMPLEMENTATIONS

Theory is useful for researchers. But, what really matters is that if its architecture works in the real world under the real constraints. Remote sensing, healthcare, and enterprise – these are the three high value domains for implementation. Also, there are some important caution factors.

7.1 Remote Sensing

Satellite imagery, Light Detection and Ranging (LiDAR), and sensor streams do not look like simple text. For processing them, specialist models are needed that understand spatial data, before any reasoning can happen. The agentic systems split the work - lightweight SLMs handle the local deductions on high-volume sensor data at the edge, while cloud LLMs handle fusion that is cross-modal, and also handles long-horizon planning [10].
Alipour Talemi et al. [14] also found that distributing work across the agents reduced propagation of errors under noisy or incomplete observations. This becomes a real problem in Earth observation. When one agent produces a bad output, others catch and correct it. And then a final decision is made. Their 2025 study [14] provided a detailed breakdown of agents and their roles in remote sensing, and how it influences perception, reasoning and decision -making.

7.2 Healthcare Systems

Hospitals cannot send the patient data directly to the cloud. Privacy rules and clinical governance does not allow that in most jurisdictions. Modular systems solve this by keeping the sensitive tasks such as patient summarisation, clinical coding, real-time monitoring, all of this inside secure local environments. These environments are handled by SLMs that have strict access controls [22]. When some case genuinely needs complex reasoning, higher-tier models get invoked in a selective way, and this helps a lot to minimize exposure. Also, Reynolds and Thornton [22] put the argument that splitting of autonomy across the lightweight agents improves accountability and trust and confidence in the clinician. This is better when compared to one opaque large model. Also, the TechRxiv healthcare survey as cited in [37]

© Author(s). This work is peer-reviewed, openly published, and permanently archived
This article is openly accessible and reusable with proper attribution.

documented the same shift happening, at a larger scale, across the real hospital deployments.

7.3 Enterprise and Industry

Irugalbandara et al. [10] showed that replacing generalized layers of LLM over rigid, underlying logical structures, with the task-matched SLM agents is beneficial. It helped to cut the operational costs by over 50%, without hurting task performance. Also, the authors of [2] cover topics such as adoption of customer support, supply chain, and intelligent automation. And the routine tasks go to the SLMs, while the exception handling and strategic decisions go to LLMs. The authors also found that multi-agent setups surpass the single-agent systems [2]. This happens when the workflows involve negotiation, escalation, cross-department coordination, etc. Modularity, or the separation of components of the system, is considered to be the cheaper and more reliable solution.

8 CHALLENGES & FUTURE DIRECTIONS

In many cases, modular agentic systems work better than the singular large models. But there is lack of confidence regarding whether they are safe, understandable, and imitable enough for the domains that need them the most.

8.1 Interpretability and Transparency

When one model makes a decision, researchers can directly ask it why. But when ten agents make a decision together, understanding what happened can become genuinely hard. Reynolds and Thornton [22] also noted this directly, that, modularity helps at the component level, but emergent behavior, like unexpected capabilities or patterns from the agent interactions can make the overall system opaque. This exists in theory, such as logs of interaction, role-wise assignment, and interpretable routing .But in reality, there are only few deployed systems that actually implement it well.

8.2 Safety and Alignment

Communicative agents for “mind” exploration of large language model society (CAMEL) [1] showed that rules about constrained dialogue reduced some safety risks. But this system only covers the cases that can be anticipated or forecasted. The harder problem is when the agents find unexpected ways to coordinate, such as feedback loops, unintended strategies, and unexpected patterns that nobody designed. Already, the alignment in single models is unsolved. And in the case of decentralised multi-agent systems it is considerably harder. In this study almost no papers were found that is tackling it seriously.

8.3 Standardization

This is the most practical problem seen, and also the most solvable one. Right now, all multi-agent system use its own message format, its own coordination logic, and its own failure handling system. So, reproducing the result and report from one lab to another is nearly impossible. Both the survey [2] and the TechRxiv healthcare review [37] ask for communication standards that is symmetrical to the network protocols, or something like HTTP is required, but for agents. Until that exists, same reinventions and coordination infrastructure will be seen, again and again.

9 CONCLUSION

This survey was started with a simple observation. The teams deploying AI in hospitals, factories, and satellite systems showed the same problems. Everytime it was observed that it was too slow, too expensive and too risky for private data. It was seen that one large general-purpose model was not the answer.

What the research shows, across three years of consistent literature, since the beginning of large scale advent of LLM-SLM models, is that the right architecture for most real deployments is a team of smaller focused models working together . Each can handle the tasks it is built for, and there are clear rules about how they communicate and what to do when they disagree. It is not elegant in a way a large language model is, but it works better in practice.

The three domains seen were remote sensing, healthcare, and enterprise. To create better systems, constraints like keeping the sensitive data local is to be rejected. It is a valuable design requirement. Researchers and users should go for a combination where edge models are run locally, and cloud models are invoked only when genuinely needed, and

© Author(s). This work is peer-reviewed, openly published, and permanently archived
This article is openly accessible and reusable with proper attribution.

there is structured coordination between them. This is a combination that helps to cut costs, reduce latency, and helps to keep the data where it belongs.

The field has moved at a surprising rate in the last four years. In 2022, this was mostly theoretical. By 2026, teams are running these systems in real hospitals and on real production lines, and getting real cost reductions. The direction is very clear. But, there is still a lack of clarity about how these systems can be made safe. For example, tracing the decisions across ten agents can be genuinely hard. In conclusion it can be said that multi-agent system is an open problem. There is no agreed standard for how agents should talk to each other. And this means that every research group is building the same infrastructure, pipelines, and backend systems from scratch. Nonetheless, these aforementioned reasons are not enough to abandon this approach. They will be the next set of problems to solve. This discussion can be ended with the understanding that this survey will be a useful starting point for researchers .

REFERENCES

1. Li, G., Hammoud, H. A. A. K., Itani, H., Khizbullin, D., & Ghanem, B. (2023). CAMEL: Communicative agents for "mind" exploration of large language model society. arXiv. https://arxiv.org/abs/2303.17760
2. Joshi, S. (2024). Review of autonomous and collaborative agentic AI and multi-agent systems for enterprise applications. International Journal of Advanced Computer Science and Applications, 15(7).
3. Williams, F., Preuss, M., & Emmerich, M. (2024). Designing efficient agentic AI: How small language models compare to large models. In Proceedings of the Genetic and Evolutionary Computation Conference Companion.
4. Shoham, Y. (1993). Agent-oriented programming. *Artificial Intelligence, 60*(1), 51–92. https://doi.org/10.1016/0004-3702(93)90034-9
5. Hosseini, S., & Seilani, H. (2024). The role of agentic AI in shaping a smart future: A systematic review. *Journal of King Saud University - Computer and Information Sciences, 36*(2), 101966. https://doi.org/10.1016/j.jksuci.2024.101966
6. Bandi, A., Kongari, B., Naguru, R., Pasnoor, S., & Vilipala, S. V. (2024). The rise of agentic AI: A review of definitions, frameworks, architectures, applications, evaluation metrics, and challenges. Journal of Artificial Intelligence Research, 1(1), 45-78.
7. Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., & Cao, Y. (2023). ReAct: Synergizing reasoning and acting in language models. *arXiv*. https://arxiv.org/abs/2210.03629
8. Sapkota, R., Roumeliotis, K. I., & Karkee, M. (2024). AI agents vs. agentic AI: A conceptual taxonomy, applications and challenges. *AI and Ethics*. https://doi.org/10.1007/s43681-024-00498-3
9. Bandi, A., Kongari, B., Naguru, R., Pasnoor, S., & Vilipala, S. V. (2024). The rise of agentic AI: A review of definitions, frameworks, architectures, applications, evaluation metrics, and challenges. Journal of Artificial Intelligence Research, 1(1), 45-78.
10. Irugalbandara, C., et al. (2024). Scaling down to scale up: A cost-benefit analysis of replacing OpenAI's LLM with open source SLMs in production. Proceedings of the 2024 ACM Symposium on Cloud Computing.
11. Belcak, P., Heinrich, G., Diao, S., Fu, Y., Dong, X., Muralidharan, S., Lin, Y. C., & Molchanov, P. (2024). Small language models are the future of agentic AI. arXiv. https://arxiv.org/abs/2411.06667
12. Nguyen, C. V., Shen, X., Aponte, R., Xia, Y., Basu, S., Hu, Z., Chen, J., Parmar, M., Kunapuli, S., Barrow, J., Wu, J., Singh, A., Wang, Y., Gu, J., Dernoncourt, F., Ahmed, N. K., Lipka, N., Zhang, R., Chen, X., … & Shi, Y. (2024). A survey of small language models. arXiv. https://arxiv.org/abs/2406.04225
13. Ren, S., Xie, C., Jian, P., Ren, Z., Leng, C., & Zhang, J. (2024). Towards scientific intelligence: A survey of LLM-based scientific agents. arXiv. https://arxiv.org/abs/2408.03549
14. Alipour Talemi, N., Boone, J., & Afghah, F. (2024). Agentic AI in remote sensing: Foundations, taxonomy, and emerging systems. IEEE Geoscience and Remote Sensing Magazine.
15. Li, J., Xu, C., Jia, L., Wang, F., Zhang, C., & Liu, J. (2024). EACO-RAG: Towards distributed tiered LLM deployment using edge-assisted and collaborative RAG with adaptive knowledge update. arXiv. https://arxiv.org/abs/2407.10849
16. Xu, R., Shi, W., Zhuang, Y., Yu, Y., Ho, J. C., Wang, H., & Yang, C. (2024). Collab-RAG: Boosting retrieval-

© Author(s). This work is peer-reviewed, openly published, and permanently archived
This article is openly accessible and reusable with proper attribution.

augmented generation for complex question answering via white-box and black-box LLM collaboration. arXiv. https://arxiv.org/abs/2407.01421
17.Patil, S. G., Zhang, T., Wang, X., & Gonzalez, J. E. (2023). Gorilla: Large language model connected with massive APIs. arXiv. https://arxiv.org/abs/2305.15334
18.Zhang, P., Zeng, G., Wang, T., & Lu, W. (2024). TinyLlama: An open-source small language model. arXiv. https://arxiv.org/abs/2401.02385
19.Liu, Z., Zhao, C., Iandola, F., Lai, C., Tian, Y., Fedorov, I., Xiong, Y., Chang, E., Shi, Y., Krishnamoorthi, R., Lai, L., & Chandra, V. (2024). MobileLLM: Optimizing sub-billion parameter language models for on-device use cases. arXiv. https://arxiv.org/abs/2402.14905
20.Wang, H., Ma, S., Dong, L., Huang, S., Wang, H., Ma, L., Yang, F., Wang, R., Wu, Y., & Wei, F. (2024). BitNet: Scaling 1-bit transformers for large language models. arXiv. https://arxiv.org/abs/2310.11453
21.Jovanović, M., & Campbell, M. (2024). Compacting AI: In search of the small language model. IEEE Potentials, 43(4), 18-24.
22.Reynolds, E. J., & Thornton, M. A. (2025). Design principles for lightweight agentic AI: Balancing autonomy, efficiency, and interpretability in small language models. Stanford University Technical Report.
23.Hao, Z., Jiang, H., Jiang, S., Ren, J., & Cao, T. (2024). Hybrid SLM and LLM for edge-cloud collaborative inference. In Proceedings of the 2024 ACM International Conference on Mobile Systems, Applications, and Services.
24.Wu, Y., Ma, L., Zhang, R., Liu, Y., Niyato, D., Tang, S., Xiong, Z., Han, Z., Yang, Z., Huang, K., Zhang, Z., & Wong, K.-K. (2024). Towards edge general intelligence: Knowledge distillation for mobile agentic AI. arXiv. https://arxiv.org/abs/2410.09582
25.Varney, E. (2024). Transistors and symphonies: Orchestrated networks of specialized small language models as a scalable alternative to monolithic LLMs. Journal of Machine Learning Research, 25(320), 1-45.
26.Derouiche, H., Brahmi, Z., & Mazeni, H. (2024). Agentic AI frameworks: Architectures, protocols, and design challenges. Journal of Computer and Communications, 12(3), 1-20.
27.Jiang, D., Ren, X., & Lin, B. Y. (2023). LLM-blender: Ensembling large language models with pairwise ranking and generative fusion. In Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics.
28.Wang, J., Wang, J., Athiwaratkun, B., Zhang, C., & Zou, J. (2024). Mixture-of-agents enhances large language model capabilities. arXiv. https://arxiv.org/abs/2406.04692
29.Chen, Y., Zhao, J., & Han, H. (2024). A survey on collaborative mechanisms between large and small language models. Journal of Artificial Intelligence Research, 79, 1-45.
30.Liu, X., Yu, H., Zhang, H., Xu, Y., Lei, X., Lai, H., Gu, Y., Ding, H., Men, K., Yang, K., Zhang, S., Deng, X., Zeng, A., Du, Z., Zhang, C., Shen, S., Zhang, T., Su, Y., Sun, H., … & Tang, J. (2024). AgentBench: Evaluating LLMs as agents. arXiv. https://arxiv.org/abs/2308.03688
31.Peng, Q., Liu, H., Huang, H., Yang, Q., & Shao, M. (2024). A survey on LLM-powered agents for recommender systems. ACM Computing Surveys.
32.Ruan, J., Chen, Y., Zhang, B., Xu, Z., Bao, T., Du, G., Shi, S., Mao, H., Li, Z., Zeng, X., & Zhao, R. (2024). TPTU: Large language model-based AI agents for task planning and tool usage. arXiv. https://arxiv.org/abs/2408.01431
33.Schick, T., Dwivedi-Yu, J., Dessì, R., Raileanu, R., Lomeli, M., Zettlemoyer, L., Cancedda, N., & Scialom, T. (2023). Toolformer: Language models can teach themselves to use tools. arXiv. https://arxiv.org/abs/2302.04761
34.Mukherjee, S., Mitra, A., Jawahar, G., Agarwal, S., Palangi, H., & Awadallah, A. (2023). Orca: Progressive learning from complex explanation traces of GPT-4. arXiv. https://arxiv.org /abs/2306.02707
35.Gunasekar, S., Zhang, Y., Aneja, J., Mendes, C. C. T., Del Giorno, A., Gopi, S., Javaheripi, M., Kauffmann, P., de Rosa, G., Saarikivi, O., Salim, A., Shah, S., Behl, H. S., Wang, X., Bubeck, S., Eldan, R., Kalai, A. T., Lee, Y. T., & Li, Y. (2023). Textbooks are all you need. arXiv. https://arxiv.org/abs/2306.11644
36.Duffy, E. B., Fernandez, D., de Waal, A., & Pesé, M. (2024). Small language models on the edge for real-world agentic systems in industry. In Proceedings of the 2024 IEEE International Conference on Edge Computing.
37.Xu, G., Li, X., Chen, Y., Duan, Y., Wu, S., Yu, A., Chiu, C.-H., Ni, J., Tang, N., Li, T. J.-J., Yuille, A., Jin, W., & Shi, Y. (2024). A comprehensive survey of AI agents in healthcare. arXiv. https://arxiv.org/abs/2408.15908

© Author(s). This work is peer-reviewed, openly published, and permanently archived
This article is openly accessible and reusable with proper attribution.