Fraunhofer FIT

Fraunhofer-Institut für Angewandte Informationstechnik FIT

WANN IST EIN MULTI-AGENTEN-SYSTEM EINE GUTE WAHL?

Warum nicht jeder Use Case mehrere Agenten braucht – und wann Multi-Agenten-Systeme sinnvoll sind.

Juni 2026

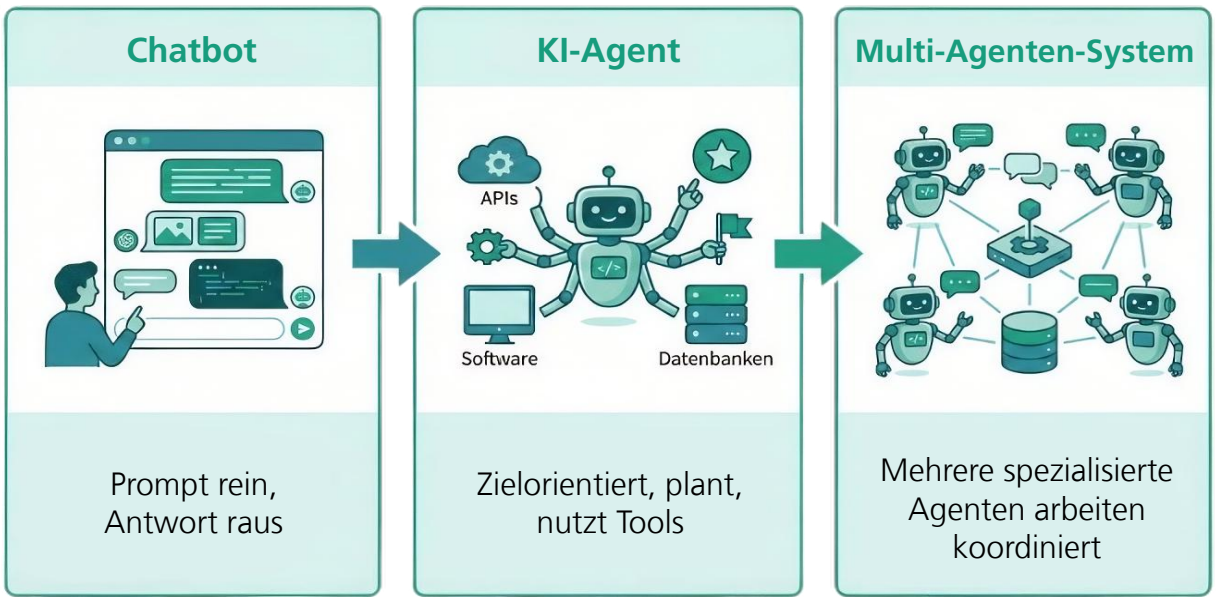
WARUM NICHT JEDER USE CASE MEHRERE AGENTEN BRAUCHT – UND WANN MULTI-AGENTEN-SYSTEME SINNVOLL SIND.

Nachdem der vorangehende Beitrag [0] die Entwicklung vom Chatbot über den KI-Agenten bis zum **Multi-Agenten-System** eingeordnet hat, stellt sich nun die Frage, wann ein Multi-Agenten-System für einen konkreten Use Case tatsächlich die richtige Wahl ist. Denn trotz des aktuellen Hypes wird vieles bereits als „Agent" bezeichnet, obwohl die tatsächliche Architektur deutlich einfacher ist. Menlo Ventures bringt das in einem Report auf den Punkt:

> ***Nur 16 % der Enterprise-Deployments qualifizieren sich als „true agents" – viele produktive Architekturen bleiben einfacher als der aktuelle Hype vermuten lässt [6].***

Als solche werden dort Systeme beschrieben, in denen typischerweise ein **Large Language Model (LLM)** Aktionen plant und ausführt, Feedback beobachtet und sein Verhalten anpasst. Mit anderen Worten: Agenten haben einen gewissen Grad an Autonomie.

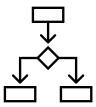
Die Beobachtung von Menlo Ventures ist ein guter Realitätscheck: Oft reicht ein klassischer Workflow, ein einzelner Agent mit Toolzugriff oder sogar eine deutlich einfachere Technik. Multi-Agenten-Systeme sind kein Selbstzweck; sie sind nur dann sinnvoll, wenn die zusätzliche Komplexität echten Mehrwert schafft.



WAS IST EIN MULTI-AGENTEN-SYSTEM?

KI-basierte Multi-Agenten-Systeme bestehen aus mehreren spezialisierten KI-Agenten, die kooperativ interagieren (vgl. [0]). So können sie komplexe Aufgaben bewältigen, bei denen **Rollenverteilung, Arbeitsteilung und Koordination** Vorteile gegenüber einem einzelnen Agenten bieten [2]. Sinnvoll sind sie besonders dann, wenn Aufgaben für einen einzelnen Agenten zu umfangreich werden oder sich spezialisierte Rollen dynamisch an den Kontext anpassen müssen. Dieser Vorteil geht jedoch mit verteilter Entscheidungs- und Problemlösungslogik sowie zusätzlichem Koordinationsaufwand einher, der zentral oder dezentral organisiert sein kann. Ergebnisse entstehen daher nicht in einem einzelnen Verarbeitungsschritt, sondern aus dem Zusammenspiel mehrerer Beiträge.

Wichtig ist die Abgrenzung: Nicht jedes agentische System ist automatisch ein Multi-Agenten-System. Auch ein einzelner Agent kann agentisch sein, wenn er ein Ziel verfolgt, Informationen aus seiner Umgebung aufnimmt, plant, handelt, Rückmeldungen auswertet und sein Verhalten anpasst [1, 4, 6]. Ein einfaches Beispiel ist ein Chatbot, der zunächst eher als Assistenzsystem gilt. Nutzt er jedoch eigenständig eine Websuche, bewertet die Ergebnisse und entscheidet über weitere Recherchen, kann er als Agent mit externem Tool eingeordnet werden [3, 4]. Multi-Agenten-Systeme gehen darüber hinaus: Sie verteilen Aufgaben, Verantwortung und Teilentscheidungen auf mehrere spezialisierten Agenten [2, 4]. Die folgende Tabelle stellt die wesentlichen Unterschiede zwischen Einzelagenten und Multi-Agenten-Systemen gegenüber.

	Kriterium	Einzelagent	Multi-Agenten-System
	Rollen und Aufgaben	Ein Agent übernimmt alle Aufgaben: Verstehen, Planen, Ausführen, Bewerten und Korrigieren.	Aufgaben, Planung, Verantwortung und Teilentscheidungen sind auf spezialisierte Agenten verteilt.
	Entscheidungs-struktur	Entscheidungen entstehen zentral in einer Instanz.	Entscheidungen entstehen kollaborativ zwischen mehreren Agenten (die finale Entscheidung kann bei einem Agenten liegen).
	Koordination und Zusammenarbeit	Keine explizite Koordination notwendig.	Abstimmung, Übergabe von Zwischenergebnissen und Konfliktlösung sind notwendig (ggf. durch zentralen Orchestrator-Agenten).
	Kontext und Kontrolle	Ein Agent hält den Gesamtkontext; Risiko von Kontextüberlastung.	Getrennte Kontexte: bessere Fokussierung, robustere Bewertung möglich.
	Geeigneter Problemtyp	Lineare, mehrstufige Aufgaben.	Offene, dynamische Probleme mit Bedarf an Kollaboration und Spezialrollen.
	Komplexität und Betriebsaufwand	Geringerer Orchestrierungs- und Monitoringaufwand.	Höherer Koordinations-, Monitoring-, Governance- und Integrationsaufwand.

NICHT JEDE AUFGABE BRAUCHT MEHRERE AGENTEN

Nach der begrifflichen Abgrenzung stellt sich nun die praktische Frage, **wann ein Multi-Agenten-System architektonisch überhaupt sinnvoll ist**. Nicht jede Aufgabe profitiert von Spezialisierung, Koordination und verteilter Entscheidungslogik. Für die Einordnung ist deshalb eine einfache, bewusst **pragmatische Dreiteilung** hilfreich.

1 **Einfache, statische Aufgaben**

2 **Komplexere, aber noch gut strukturierbare Aufgaben**

3 **Offene, dynamische und kollaborative Aufgaben**

Für die erste Kategorie sind Multi-Agenten-Systeme meist überdimensioniert; in der zweiten reichen oft Einzelagenten, Prompt Chaining oder klar orchestrierte Workflows [3, 4]. **Ihre eigentlichen Stärken zeigen Multi-Agenten-Systeme vor allem bei offenen, dynamischen und kollaborativen Problemen**: also dann, wenn mehrere spezialisierte Rollen gemeinsam mehr leisten als ein einzelner Generalist [1, 2].

•1

Eine aktuelle Studie ergänzt diese Heuristik um einen wichtigen Punkt: **Ob zusätzliche Spezialisierung tatsächlich Vorteile bringt, hängt nicht nur vom Use Case, sondern auch von der Stärke des eingesetzten Basismodells ab.** Mit leistungsfähigeren Modellen schrumpft der Vorsprung von Multi-Agenten-Systemen gegenüber Einzelagenten; deshalb können auch hybride Ansätze sinnvoll sein [9].

DIE WICHTIGSTE HEURISTIK: WÜRDE EIN MENSCH EIN KLEINES TEAM ZUSAMMENSTELLEN?

Ein Multi-Agenten-System ist dann besonders sinnvoll, wenn ein Mensch spontan sagen würde: „Für diese Aufgabe brauche ich eigentlich ein kleines Team aus Spezialisten.“ Diese Analogie bietet eine pragmatische Orientierung, auch wenn sie keine harte Regel ist.

In der Softwareentwicklung wird Rollenteilung besonders anschaulich [3]: Product Owner, Requirements Engineer, Backend-Entwickler, Frontend-Entwickler und Test Engineer bringen jeweils andere Perspektiven, Daten, Werkzeuge und Verantwortlichkeiten mit.

Spezialisierte Agenten übernehmen Test- und Entwicklungsschritte: Sie automatisieren Tests, orchestrieren deren Ausführung, analysieren Fehlerbilder und stoßen weitere Korrektur-, Verarbeitungs- und Coding-Schritte an. Danach kann der Zyklus erneut beginnen: Ergebnisse aus Testing, Deployment oder Monitoring fließen in weitere Anpassungen ein, deren Auswirkungen erneut geprüft und bewertet werden [7]. In geeigneten Multi-Agenten-Architekturen können weitere Iterationen, nächste Schritte oder Abbruchkriterien auch verteilt entschieden werden – etwa durch Abstimmung, Verhandlung oder explizite Bewertungsmechanismen [2]. Genau hier zeigt sich der eigentliche Nutzen eines Multi-Agenten-Systems: Es bildet nicht nur mehrere Schritte ab, sondern Zusammenarbeit, Spezialisierung und iterative Abstimmung.

Übertragen auf KI – Gute Use Cases beruhen auf natürlicher Zusammenarbeit: Gute Use Cases für Multi-Agenten-Systeme sind solche, die sich natürlich als Zusammenarbeit mehrerer Rollen modellieren lassen. Die verschiedenen Rollen bringen idealerweise eigene Fähigkeiten, Werkzeuge, Datenzugriffe oder Bewertungsmaßstäbe mit. Der Vorteil liegt dann in Arbeitsteilung, Spezialisierung und Koordination.

Im Unternehmenskontext kann Agententrennung organisatorisch sinnvoll sein: Wenn unterschiedliche Abteilungen jeweils eigenes Domänenwissen oder eigene Datenquellen verantworten, können spezialisierte Agenten diese fachlichen Einheiten sauber abbilden und in ein gemeinsames Multi-Agenten-System einbringen. Der organisatorische Nutzen liegt dann darin, dass Wissen und Verantwortung dort verankert bleiben, wo sie fachlich und organisatorisch hingehören [4].

WORAN MAN GUTE MULTI-AGENTEN-USE-CASES ERKENNT

Aus der zuvor beschriebenen Grundidee, dass Multi-Agenten-Systeme vor allem dann sinnvoll sind, wenn sich ein Use Case als Zusammenarbeit spezialisierter Rollen mit eigenen Fähigkeiten, Werkzeugen, Datenzugriffen oder Bewertungsmaßstäben modellieren lässt, lassen sich einige recht klare Eigenschaften ableiten [1, 2]. **Ein Use Case ist besonders gut für ein Multi-Agenten-System geeignet, wenn möglichst viele der folgenden Punkte erfüllt sind**:

Es wird echte Spezialisierung gebraucht:
Unterschiedliche Teilprobleme erfordern verschiedene Fähigkeiten, Werkzeuge oder Bewertungsmaßstäbe – im Unternehmenskontext oft auch unterschiedliches Domänenwissen oder Verantwortlichkeiten.

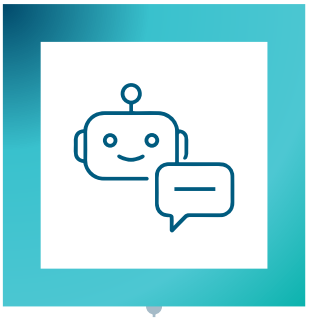
Koordination und Abstimmung sind nötig:
Koordination und Abstimmung sind nötig: Zwischenergebnisse müssen übergeben, Aufgaben sinnvoll verteilt, Konflikte aufgelöst und Prioritäten bei Bedarf neu gesetzt werden. Beiträge verschiedener Rollen müssen parallel erbracht oder sinnvoll aufeinander abgestimmt werden. Ziel ist ein konsistentes Gesamtergebnis, in dem unterschiedliche Perspektiven produktiv zusammenwirken.

Das Problem ist offen, dynamisch und von Unsicherheit geprägt:
Neue Informationen tauchen auf, Hypothesen müssen verworfen und Entscheidungen laufend neu bewertet werden.

Unterschiedliche Tools, Datenquellen oder Denkweisen müssen kombiniert werden:
Unterschiedliche Tools, Datenquellen oder Denkweisen müssen kombiniert werden: Dazu gehören etwa APIs, Wissensbestände, kreative Entwürfe, Critic- oder Reviewer-Rollen zur Prüfung fremder Zwischenergebnisse oder formale Validierung.

Die zusätzliche Komplexität der Arbeitsteilung sollte einen erkennbaren Nutzen bringen:
Die zusätzliche Komplexität der Arbeitsteilung sollte einen erkennbaren Nutzen bringen: Mehr Orchestrierung, Kommunikation und Governance lohnen sich nur, wenn die Aufteilung tatsächlich zu besseren Ergebnissen oder robusteren Prozessen führt.

WARUM DIE AUFTEILUNG AUCH TECHNISCH SINNVOLL SEIN KANN

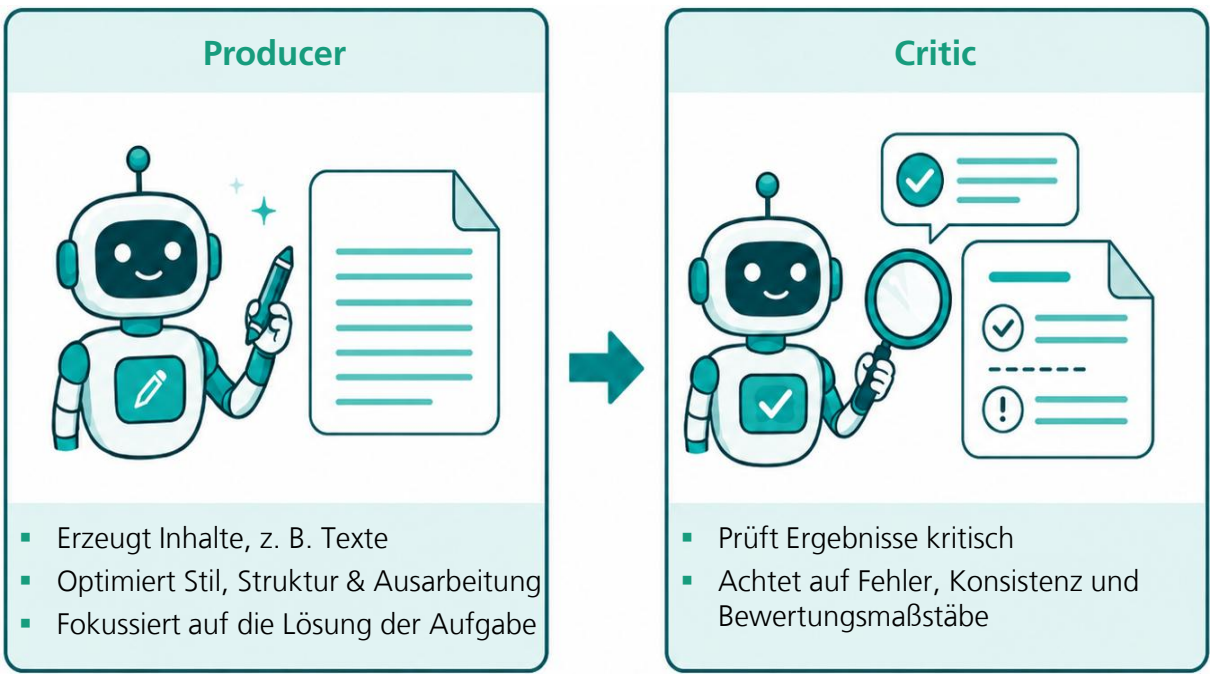


Ein zentraler technischer Grund für die Zerlegung komplexer Aufgaben in spezialisierte Rollen liegt darin, dass ein einzelnes LLM sonst schnell zu viele Funktionen gleichzeitig übernehmen muss: verstehen, planen, ausführen, Zwischenergebnisse bewerten und sich selbst korrigieren. Multi-Agenten-System verteilen diese Funktionen auf mehrere Rollen. In solchen Situationen kann eine saubere Rollentrennung helfen, Risiken wie Kontextüberlastung, Drift in Zwischenständen, Fehler-fortpflanzung und Halluzinationen zu reduzieren [1, 3, 4]. Zudem verringert sie die Konkurrenz um Aufmerksamkeit und Kontext im selben Modell und kann die Systementwicklung vereinfachen, da mehrere kleinere, klar abgegrenzte Komponenten oft leichter zu entwerfen, zu testen und zu warten sind als ein einzelnes, hochkomplexes System.

Technisch geht es damit vor allem um die Entkopplung konkurrierender Optimierungsaufgaben sowie um Aufmerksamkeits- und Kontextökonomie. Unterschiedliche Arbeitsschritte stellen unterschiedliche Anforderungen an Informationen, Ziele und Bewertungskriterien. Werden Planung, Ausführung und Bewertung in einem einzigen Agenten und demselben Kontextfenster gebündelt, konkurrieren diese Signale. Hohe Genauigkeit profitiert häufig von einem fokussierten, relevanten und gut strukturierten Kontext; Kontext-Engineering lenkt die „begrenzte Aufmerksamkeit“ des Modells gezielt auf das für den jeweiligen Schritt Relevante [1, 3, 4]. Werden Aufgaben dagegen auf getrennte Agenten verteilt, lässt sich dieser Kontext gezielter zuschneiden. Das macht die Verarbeitung stabiler und erhöht die Wahrscheinlichkeit, dass das Modell konsistent und präzise arbeitet.

WARUM DIE AUFTEILUNG AUCH TECHNISCH SINNVOLL SEIN KANN

Besonders anschaulich wird das bei einem **Producer-Critic-Modell**, etwa wenn ein Agent einen Text erzeugt und ein zweiter ihn kritisch prüft. Der erzeugende Agent optimiert auf Stil, Struktur und inhaltliche Ausarbeitung, der prüfende Agent auf Fehlerfindung, Konsistenz und Bewertungsmaßstäbe. Der Vorteil liegt nicht nur in der Arbeitsteilung, sondern auch darin, dass ein getrenntes Producer-Critic-Modell oft robuster ist als bloße Selbstreflexion [3, 4]. Der prüfende Agent betrachtet das Ergebnis mit einer frischen Perspektive und einer eigenen Rolle; dadurch wird gerade jener kognitive Bias reduziert, der entstehen kann, wenn ein Modell seine eigene Ausgabe bewertet.



Die Aufteilung ist also kein Selbstzweck, sondern ein Mittel, um komplexe Aufgaben robuster, kontrollierbarer und besser prüfbar zu gestalten. Das gilt nicht, weil Selbstprüfung grundsätzlich unmöglich wäre, sondern weil dieselbe Instanz in einem gemeinsamen Kontext widersprüchliche Rollen oft weniger stabil hält als getrennte, spezialisierte Komponenten [1].

WANN IST EIN MULTI-AGENTEN-SYSTEM NICHT DIE BESTE WAHL?

Die vielleicht wichtigste architektonische Reife zeigt sich nicht darin, möglichst schnell Multi-Agenten-Systeme einzusetzen, sondern darin, **sie nicht einzusetzen, wenn einfachere Muster ausreichen:**

Wenn die Aufgabe einfach oder einmalig ist: Für klar umrissene oder einmalige Aufgaben sind Agenten oft „overkill". Ein Skript, eine API-Integration oder klassische Automatisierung ist dann meist günstiger, robuster und leichter zu betreiben.

Wenn der Prozess linear und gut zerlegbar ist: Mehrstufige, aber lineare Probleme brauchen oft keine kollaborativen Agenten. Hier ist Prompt Chaining meist die passendere Architektur.

Wenn nur ein Spezialtool fehlt: Wenn das Problem im Kern nicht in fehlender Rollenvielfalt besteht, sondern nur darin, dass ein Modell externe Daten oder Funktionen benötigt, reicht häufig ein einzelner Agent mit Toolzugriff.

Wenn die Aufgabe zwar anspruchsvoll ist, aber keine Teamarbeit braucht: Ein Problem kann komplex sein und trotzdem besser von einem planenden Einzelagenten gelöst werden – etwa dann, wenn es vor allem um strukturierte Planung, schrittweise Ausführung und iterative Selbstkorrektur geht, aber nicht um echte Spezialisierung verschiedener Rollen.

i

Die Faustregel lautet also: **Nicht jede komplexe Aufgabe ist automatisch ein Multi-Agenten-Problem.** Manche Aufgaben sind komplex, aber zentral lösbar. Andere sind komplex, aber linear. Und wieder andere sind komplex, offen und kollaborativ – genau dort beginnt der natürliche Einsatzbereich von Multi-Agenten-Systemen.

WANN IST EIN MULTI-AGENTEN-SYSTEM NICHT DIE BESTE WAHL?

Eine harte Obergrenze für Multi-Agenten-Systeme gibt es derzeit nicht. Praktisch endet ihr sinnvoller Einsatz dort, wo zusätzlicher Koordinationsaufwand, mehr Latenz und höhere Kosten den Gewinn aus Spezialisierung und Arbeitsteilung wieder aufzehren. Das muss jedoch keine binäre Entscheidung zwischen Einzelagent und Multi-Agenten-System sein: Hybride Architekturen können mit einem einfachen Agenten oder Workflow starten und nur bei Bedarf Sub-Agenten hinzuziehen oder an ein Multi-Agenten-System eskalieren. Gerade bei stark sequenziellen Aufgaben kann zusätzliche Multi-Agenten-Koordination die Leistung sogar verschlechtern. Deshalb sollte man in der Praxis die niedrigste Architekturstufe wählen, die die Anforderungen zuverlässig erfüllt, und zusätzliche Agenten nur dort ergänzen, wo Spezialisierung oder Kooperation einen spürbaren Vorteil bringen [3, 4, 5, 9].

Die folgende Grafik fasst die zentralen Kriterien dafür zusammen, wann ein Multi-Agenten-System für einen Use Case sinnvoll ist – und wann eine einfachere Architektur meist die bessere Wahl bleibt.

	Geeignet für Multi-Agenten-Systeme		Ungeeignet für Multi-Agenten-Systeme
✓	**Spezialisierte Aufgaben** Beispiel: Softwareentwicklung und getrennte Rollen.	✗	**Einfach & einmalig** Beispiel: Datenexport und-transformation.
✓	**Koordination und Abstimmung nötig** Beispiel: Mehrere Agenten müssen gemeinsam eine Entscheidung treffen.	✗	**Vorgegebener Prozess mit aufeinanderfolgenden Schritten** Beispiel: Prozess mit klaren Schritten.
✓	**Dynamisches und unsicheres Umfeld** Beispiel: Nicht-deterministische Prozesse mit stetig neuem Informationsinput.	✗	**Komplex, aber ohne Teamarbeit** Beispiel: Automatisierte Berichterstellung aus einer Datenquelle.
✓	**Unterschiedliche Tools kombinieren** Beispiel: Analyse, Recherche und Validierung über verschiedene APIs.	✗	**Nur ein Spezial-Tool fehlt** Beispiel: Web Scraping mit einem Einzelagenten.
✓	**Mehrwert durch Aufteilung der Komplexität** Beispiel: Generator- und Review-Agent zur Qualitätssicherung.	✗	**Mehr Kosten als Nutzen** Beispiel: Aufwändige Koordination und nur minimal bessere Ergebnisse.

Frage: Würde man für den Use Case ein Team aus Spezialisten zusammenstellen?

DER PREIS ZUSÄTZLICHER AGENTEN: KOMPLEXITÄT, KOSTEN UND GOVERNANCE

Multi-Agenten-Systeme sollte man weniger als „eine KI mit vielen Untermodulen“ verstehen, sondern eher als organisierte Zusammenarbeit spezialisierter Rollen. Grundsätzlich lassen sich **zentrale, dezentrale und hybride Koordinationsmuster** unterscheiden:

Wie arbeiten Agenten in einem Multi-Agenten-System zusammen?

Ein Supervisor kann Aufgaben planen, verteilen, Ergebnisse zusammenführen und Konflikte auflösen, schafft aber auch Abhängigkeiten und mögliche Engpässe [1].

Dezentrale Ansätze verteilen Kommunikation und Entscheidung stärker auf die beteiligten Agenten, etwa über direkte Abstimmung, Konsensmechanismen oder marktartige Verfahren. Sie sind dadurch flexibler, aber auch koordinationsintensiver [1, 2].

In der Praxis liegen viele Architekturen zwischen diesen Polen. Die konkrete Ausgestaltung solcher Koordinationsmuster wird in einem folgenden Beitrag näher betrachtet.

So attraktiv das Konzept klingt: Multi-Agenten-Systeme bringen spürbar mehr Komplexität mit sich – und damit tendenziell auch höhere Kosten. Diese Komplexität zeigt sich nicht nur in Orchestrierung, Monitoring und Governance, sondern auch in höheren Security-Anforderungen und eigenen Fehlerbildern: Einzelne kritische Agenten können die Gesamtleistung begrenzen, umfangreiche Zwischenergebnisse nachgelagerte Agenten überlasten und Fehler können sich über mehrere Agentenschritte hinweg fortpflanzen [9]. Außerdem steigt der Aufwand, Ergebnisse nachvollziehbar und verlässlich zu machen. Deshalb sollte die Architekturentscheidung nicht vom Trend, sondern vom Use Case ausgehen.

Entscheidend ist nicht die Frage „Wie setze ich Multi-Agenten ein?“, sondern: „Erzeugt Arbeitsteilung hier wirklich einen Mehrwert, den ich mit einfacheren Mitteln nicht genauso gut erreichen kann?“ Ob ein Multi-Agenten-System zum Einsatz kommen sollte und wie groß oder komplex es sinnvollerweise ausfällt, ist letztlich eine Abwägung zwischen fachlichem Nutzen auf der einen sowie zusätzlicher Komplexität und Kosten auf der anderen Seite. Der Einsatz eines komplexeren Multi-Agenten-Systems ist nur dann gerechtfertigt, wenn er einen Vorteil bietet, der mit einfacheren Architekturen nicht in vergleichbarer Weise erreichbar ist.

AUSBLICK: MULTI-AGENTEN-SYSTEME & GREEN AI

Mehr Agenten ≠ mehr Nutzen

KI-Systeme im Allgemeinen und Multi-Agenten-Systeme im Besonderen gelten häufig als ressourcenintensiv und damit als potenziell wenig nachhaltig. Dieser Einwand ist berechtigt: Zusätzliche Agenten bedeuten meist mehr LLM-Inferenzaufrufe, höhere Kosten und eine stärkere Emissionswirkung. Zugleich greift eine pauschale Bewertung zu kurz, weil sie den Nutzen solcher Systeme ausblendet, der in verschiedenen Anwendungsfeldern erheblich sein kann. Das Risiko ineffizienter Multi-Agenten-Systeme bleibt jedoch real: Werden zu häufig große Modelle eingesetzt oder Kommunikations- und Entscheidungs-Mechanismen ungünstig gestaltet, kann es zu übermäßiger Abstimmung, unnötigen Verzögerungen und hohem Ressourcenverbrauch kommen. Eine aktuelle Studie zu agentischen Coding-Aufgaben zeigt, wie stark dieser Effekt ausfallen kann: Dort lag der Tokenverbrauch bis zu 1.000-mal höher als bei den betrachteten Code-Reasoning- und Code-Chat-Szenarien, maßgeblich getrieben durch Input-Token und mitgeführten Kontext [10]. Mehr Token führten dabei nicht automatisch zu besseren Ergebnissen.

Green AI beginnt bei der Architektur

Gerade deshalb ist die konkrete Architektur entscheidend. Weil Teilaufgaben auf spezialisierte Rollen verteilt werden, können unterschiedliche Agenten mit unterschiedlich großen Modellen arbeiten [3, 4]: Einfache Orchestrierungs-, Prüf- oder Hilfsagenten brauchen oft kein sehr leistungsfähiges, großes Basismodell, während für anspruchsvolle juristische Recherche oder das Verfassen komplexer Dokumente ein leistungsfähigeres Modell sinnvoll sein kann. Wird diese Modellwahl intelligent getroffen, lassen sich unnötige Token- und Inferenzkosten reduzieren. Auch die Kontext- und Speicherarchitektur der einzelnen Agenten sowie die Frage, wie viel Kontext jeweils sinnvoll ist, müssen sorgfältig gewählt werden. Letztlich kann ein gut gestaltetes Multi-Agenten-System im Einzelfall trotz zusätzlicher Koordination effizienter und potenziell auch mit geringerer Emissionswirkung arbeiten als ein monolithischer Ansatz, der jede Teilaufgabe mit demselben großen Modell bearbeitet. Unter Nachhaltigkeitsgesichtspunkten kommt es daher vor allem auf die konkrete Systemgestaltung an. Green AI ist für Multi-Agenten-Systeme deshalb kein Widerspruch, sondern vor allem eine Frage guter Architektur [8].

FAZIT

Nicht alles ist ein Multi-Agenten-System-Use-Case

Multi-Agenten-Systeme sind kein pauschales Upgrade für jeden KI-Anwendungsfall, sondern eine spezifische Antwort auf Probleme, bei denen Spezialisierung, Koordination und Kontexttrennung einen echten architektonischen Vorteil bringen. Sie lohnen sich vor allem dann, wenn eine Aufgabe in klar unterscheidbare Rollen zerfällt und ein einzelnes Modell sonst zu viele Ziele, Funktionen und Kontrollaufgaben gleichzeitig tragen müsste. Ob sie die richtige Wahl sind, ist letztlich eine Abwägung zwischen zusätzlichem fachlichem Nutzen auf der einen und zusätzlicher Komplexität sowie höheren Kosten auf der anderen Seite. Wo diese Voraussetzungen fehlen, ist eine einfachere Architektur meist die bessere Wahl – robuster, nachvollziehbarer und wirtschaftlicher.

QUELLEN

[0] Fetzer, D., Gimpel, H., Jung, C., & Strasser, S. (2026). *Vom Chatbot zum Multi-Agenten-System: Wie aus reaktiver Inhaltsproduktion koordinierte Autonomie entsteht* [Knowledge Nugget]. Fraunhofer-Institut für Angewandte Informationstechnik FIT. https://www.wi.fit.fraunhofer.de/content/dam/fit/witschaftsinformatik/dokumente/01_AWA_Chatbot-Agents-MAS.pdf

[1] Gullí, A. (2025). *Agentic design patterns: A hands-on guide to building intelligent systems*. Springer. https://doi.org/10.1007/978-3-032-01402-3

[2] Huang, K. (Hrsg.). (2025). *Agentic AI: Theories and practices*. Springer. https://doi.org/10.1007/978-3-031-90026-6

[3] Schluntz, E., & Zhang, B. (2024, 19. Dezember). *Building effective agents*. Anthropic Engineering. https://www.anthropic.com/engineering/building-effective-agents

[4] Kittel, C., Siemens, C., et al. (2026, 12. Februar). *AI agent orchestration patterns*. Microsoft Azure Architecture Center. https://learn.microsoft.com/en-us/azure/architecture/ai-ml/guide/ai-agent-design-patterns

[5] Kim, Y., & Liu, X. (2026, 28. Januar). *Towards a science of scaling agent systems: When and why agent systems work*. Google Research. https://research.google/blog/towards-a-science-of-scaling-agent-systems-when-and-why-agent-systems-work/

[6] Tully, T., Redfern, J., Das, D., & Xiao, D. (2025, 9. Dezember). *2025: The state of generative AI in the enterprise*. Menlo Ventures. https://menlovc.com/perspective/2025-the-state-of-generative-ai-in-the-enterprise/

[7] Reply. (2025, 15. Mai). *Reply launches Silicon Shoring: An AI-powered software delivery model to optimise and automate the entire software development life cycle*. Reply News. https://www.reply.com/en/newsroom/news/reply-launches-silicon-shoring-an-ai-powered-software-delivery-model-to-optimise-and-automate-the-entire-software-development-life-cycle

[8] Rakhunathan, S. (2025, 10. Juni). *Sustainable AI design for workloads on Azure*. Microsoft Learn. https://learn.microsoft.com/en-us/azure/well-architected/sustainability/sustainable-ai-design

[9] Gao, M., Li, Y., Liu, B., Yu, Y., Wang, P., Lin, C.-Y., & Lai, F. (2025). *Single-agent or multi-agent systems? Why not both?* arXiv. https://doi.org/10.48550/arXiv.2505.18286

[10] Bai, L., Huang, Z., Wang, X., Sun, J., Mihalcea, R., Brynjolfsson, E., Pentland, A., & Pei, J. (2026). *How do AI agents spend your money? Analyzing and predicting token consumption in agentic coding tasks*. arXiv. https://doi.org/10.48550/arXiv.2604.22750

Fraunhofer-Institut für Angewandte Informationstechnik FIT

Generative AI Lab

Alter Postweg 101, 86159 Augsburg
www.fit.fraunhofer.de
https://www.linkedin.com/company/fraunhofer-fit-generative-ai-lab/

Ansprechpersonen:

Prof. Dr. Henner Gimpel
henner.gimpel@fit.fraunhofer.de
https://www.linkedin.com/in/henner-gimpel/

Prof. Dr. Manfred Schoch
manfred.schoch@fit.fraunhofer.de
https://www.linkedin.com/in/manfred-schoch-42257877/

Dieses Knowledge Nugget ist im Rahmen des Forschungsprojekts Agentic Work Automation (AWA) entstanden

Finanzielle Förderung:

Bayerische Transformations- und Forschungsstiftung

Projektpartner:

Ancud IT, DATEV, Fraunhofer FIT, Maisel's Weisse, soffico

Mitwirkende an diesem Knowledge Nugget:
Frank Eichinger[1], Philip Wedelich[1], Dominik Fetzer[2], Carolin Jung[2], Michael Kraus[2], Manfred Schoch[2]
[1]DATEV eG, [2]Fraunhofer FIT