Von ChatGPT zu Agentic AI
Der rote Faden der KI-Entwicklung

INNOVATE. INTEGRATE. EXCEED.

Ivan Herreros Alonso - inovex GmbH
inovex Agentic AI Meetup - Karlsruhe, 20.3.2026

inovex

inovex

November 2022 - Heute

- 3.5 Jahre ChatGPT – von Chat Interfaces zu autonomen Agenten
- Eine rasante Entwicklung – aber kein Zufall
- Wie sind wir hier angekommen?
  → Wissenschaft + Engineering
- Wohin geht die Reise noch?
  → Welche sind die relevanten Fragen?

inovex

Die Wissenschaft

inovex

1. Von Monolog ...

... zum Dialog

inovex

“Conversation” als Document Completion

- LLMs waren immer Document Completion Maschinen
  → Input: Dokument-Anfang. Output: Wahrscheinlichste Fortsetzung

- Der Trick: Das Dokument als Theaterstück strukturieren
  → Figur 1 (Human) fragt, Figur 2 (AI) antwortet

inovex

Das Theaterstück

```
User: Hello, how are you?
Assistant: I’m doing well, thank you! How can I help you
today?
User: What is the capital of France?
Assistant:
```

inovex

Das Theaterstück

```
User: Hello, how are you?
Assistant: I’m doing well, thank you! How can I help you
today?
User: What is the capital of France?
Assistant: The capital of France is Paris.
```

- Das LLM schreibt im *Theaterstück* weiter
- Mit einer ‚Stop-Generation‘-Regel plus Templating war es einfach, eine PoC-Chat-Anwendung zu entwickeln

inovex

Das Theaterstück

```
User: Hello, how are you?
Assistant: I’m doing well, thank you! How can I help you
today?
User: What is the capital of France?
Assistant: The capital of France is Paris.
User: Tell me more about it.
Assistant:
```

inovex

Das Theaterstück

```
User: Hello, how are you?
Assistant: I'm doing well, thank you! How can I help you
today?
User: What is the capital of France?
Assistant: The capital of France is Paris.
User: Tell me more about it.
Assistant: Paris is the largest city in France...
```

- Das LLM "spricht" nicht – es vervollständigt ein Dokument, das wie ein Gespräch aussieht
- Die Nützerin bekommt die Erfahrung, mit einer KI zu sprechen
- ChatGPT hat diese Idee per Reinforcement-Learning perfektioniert

inovex

2. Aus dem Dokument in die Welt

inovex

Aus dem Dokument hinaus in die Welt

- Das LLM lebt immer noch im Dokument
- Aber LLM und Backend konnten ein “*Geheimcode*” bestimmen
  → LLM schreibt: <function>get_weather("Barcelona")</function>
  → Backend erkennt das Muster und führt die Funktion aus
- LLMs konnten außerhalb des Dokuments interagieren

inovex

Tool Calling aus LLM-Perspektive

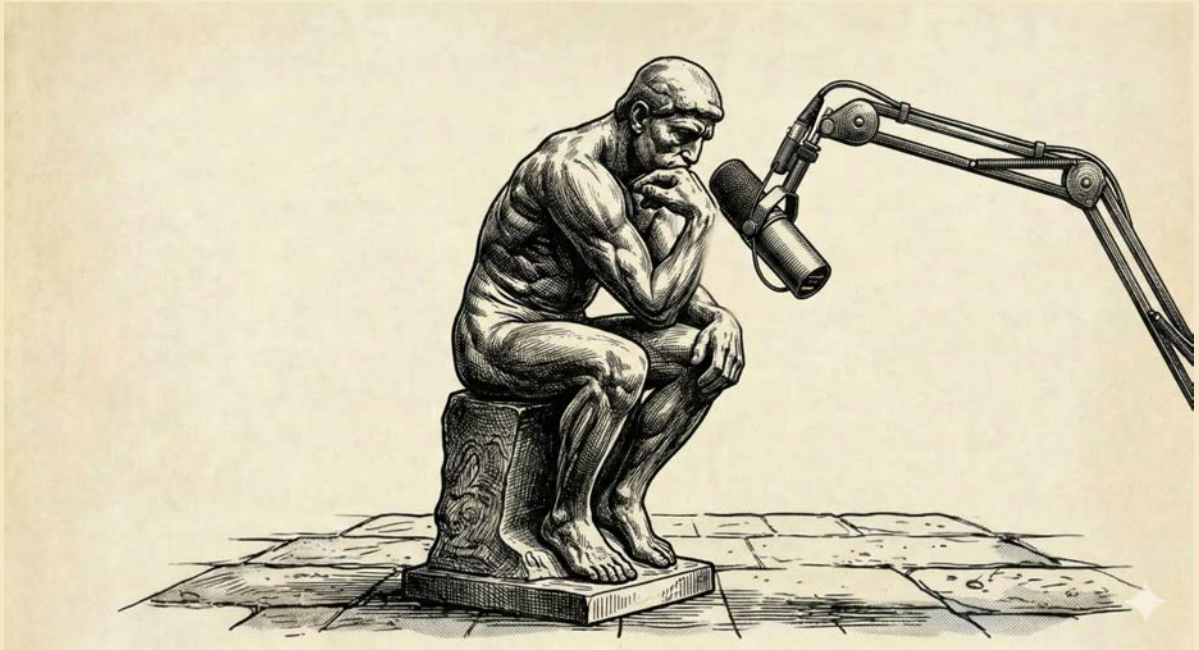
```
User: "What's the weather in Barcelona?"

LLM generates:
{
  "function": "get_weather",
  "arguments": {
    "city": "Barcelona", "country": "ES"
  }
}
System executes -> returns: {"temp": 24, "condition": "sunny"}

LLM generates:
"The weather in Barcelona is currently 24°C and sunny."
```

(Typisches Beispiel, aber auch ein schlechtes Beispiel für Tool Calling)

Durch “Function Calling” wurde die ganze digitale Welt fürs LLM potentiell erreichbar

inovex

3. Laut Denken



inovex

Reasoning-Loud: Ein Puffer zum Denken

- Zwei Szenarien
  1. 2 + 2 = 4? Antworte ja/nein
  2. <20 Seiten legales Dokument>. Ist dieser Arbeitsvertrag fair und sollte ich ihn unterschreiben? Antworte ja/nein
- Ursprüngliche LLMs verfügten für jede Frage über das gleiche “Thinking Budget”

Reasoning-Loud: Ein Puffer zum Denken

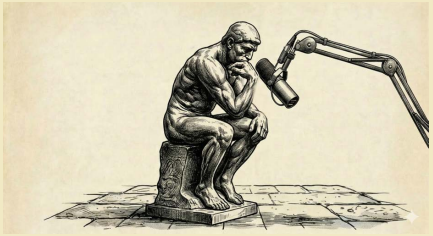
- Der Trick: Nicht das Modell ändern – sondern ihm Raum zum Denken geben
  - Erst Chain-of-Thought Prompting. “Solve it step-by-step”
  - Später “Reasoning”-Modelle (Sept. 2024)

“Transformers (d.h., LLMs) brauchen tokens zum Denken” (Karpathy, 2023)

inovex

Bausteine zum Agentic AI

Sprechen



Denken

Handeln

Mit diesen Fähigkeiten war es keine Frage **ob,** sondern **wann**

inovex

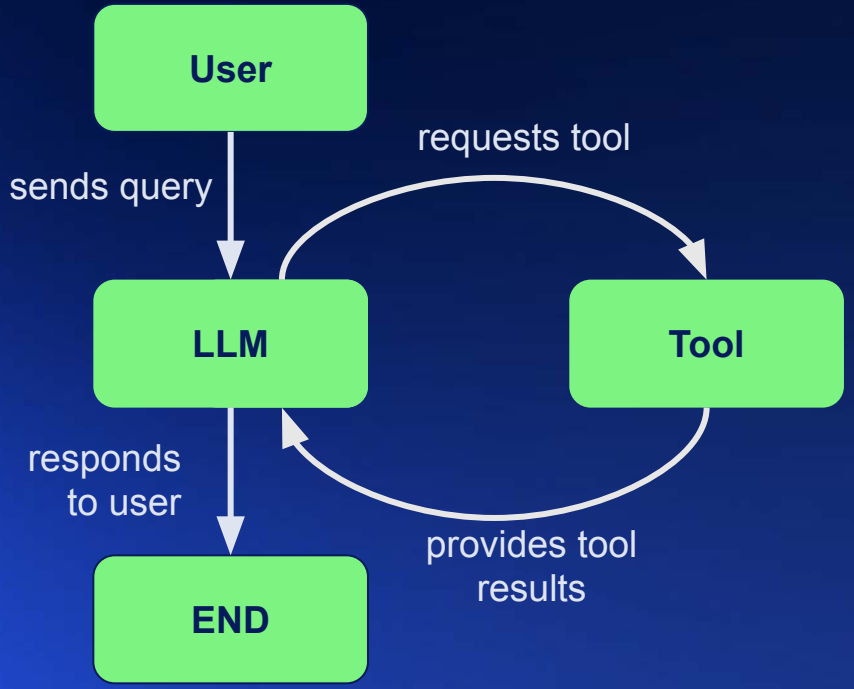
Seit wann wissen wir, dass alle Zutaten zur Agentic AI vorhanden sind?

"*ReAct: Synergizing Reasoning and Acting in Language Models*" (Yao et al., 2022)

- Wissenschaftliches Paper, das die Möglichkeit “Agentic loops” zu implementieren zur besseren Aufgabenlösung einführt
- Sehr prägender Effekt in der LLM-Welt

Das ReAct Agent

- ReAct führt das “Agentic-Loop”-Konzept ein
- Das LLM könnte um einen Query zu beantworten, iterieren, und so viele Tools wie nötig aufrufen



Das ReAct Paper kam *vor* dem Release vom chatGPT

inovex

Wenn 2023 schon alles bekannt war, warum sprechen wir erst heute über Agentic AI?

- **LLM-seitig**: Modelle waren noch nicht stark genug
- **Backend-seitig**: Es fehlten die richtigen Tools, Architecture Patterns, Integration Standards

inovex

Die Optimierungen

inovex

Zwei Kategorien von Optimierungen

Foundation Tools

- Adressieren fundamentale LLM-Limitierungen oder nutzen fundamentale LLM-Fähigkeiten
- Voraussetzung für viele Agentic Systeme

Integrationen mit Drittsystemen

- Wie verbinden wir LLMs mit der Unternehmens-Welt?
  → Vom ersten Ansatz (RAG) zur sich durchsetzenden Lösung (MCP)

Foundation Tools

- Tools, die entweder grundlegende LLM-Limitierungen überwinden oder LLM-Fähigkeiten nutzen
- Früh als sehr relevant aus User-Perspektive erkannt

Web Search

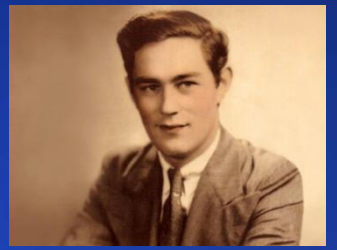
Überwindet das Cutoff-Problem

Code-Ausführung

Nutzt LLM-Fähigkeit in Programmiersprachen

Das "Anterograde Amnesia" Problem

- LLM Lifecycle: Training vs. Inference
  - “**Training**”: Knowledge wird in Weights kodiert
  - “**Inference**”: Weights produzieren Outputs, werden aber nicht weiter modifiziert
- **LLMs haben ein *Cutoff Date***
- Sie funktionieren genau wie der Patient H.M.



Web-Search Tool: partielle Lösung des Cutoffs Problem

- Das LLM kann aktuelle Informationen abrufen
- Daher können sie Fragen beantworten so wie “*Gibt’s Streik beim Nahverkehr morgen in Karlsruhe*”?
- Aber
  → “Inference” schreibt weiterhin keine neuen Infos in Weights
  → Das "Anterograde Amnesia" Problem bleibt bestehen
  → LLMs lernen nicht von unseren Interaktionen – höchstens können sie “Notizen machen” (wie der Patient H.M.)

Code-Ausführung Tool

- Dieses Tool erweitert die LLM Fähigkeiten indem es Code schreiben und selbst-ausführen kann
- Warum ist Coding eine grundlegende Fähigkeit von LLMs?
  → Code ist im Trainingskorpus von LLMs überrepresentiert
- Folge: LLMs wurden von Anfang an sehr gut in viele Programmiersprachen
- Was das erlaubt:
  → Datenanalyse & -visualisierung
  → Datenbearbeitung

Code-Ausführung: Open-ended Tool

get_weather()
Feste Funktion
Behebt einen einzelnen Use-Case
Begrenzte Flexibilität

Code-Ausführung
Universal & flexibel
Unendlich viele Use-Cases
Nur durch LLM-Fähigkeiten begrenzt

LLM-Capability-Bound: Tool-Performance skaliert mit LLM-Verbesserungen

Neue Use Cases werden ermöglicht ohne das Tool weiterzuentwickeln

Die Frage der Integration mit Drittsystemen

- **Wie nutze ich ein LLM mit meinen eigenen Daten?**

- Die 2023-24 Lösung: Retrieval-Augmented Generation (RAG)
  1. User stellt eine Frage
  2. Eine Datenbank wird durchsucht → Kleine Text-Fragmente werden zurückgegeben
  3. Fragmente + Frage werden zu einem Prompt zusammengebaut
  4. LLM antwortet auf Basis des ad-hoc Kontexts

- Auch eine Antwort auf das Anterograde-Amnesia-Problem
  → aber auf der Domain-Achse, nicht der Zeit-Achse

inovex

RAG — 2023 unverzichtbar, 2026 ein Sonderfall

- 2023: 4K–16K tokens
  → man musste gezielt die relevantesten Fragmente auswählen

- 2026: 200K–1M tokens
  → diese Einschränkung besteht kaum noch

- Aber selbst 2023/24 fühlte sich die Architektur irgendwie falsch an
  → RAG = Daten (teilweise) in die KI-Plattform kopieren
  → Ergebnis: 90% Infra/Data Engineering, 10% KI
  → Die Challenge bei RAG war eher Data Management als GenAI

inovex

Die 2025-Lösung: Model Context Protocol (MCP)

- Von Anthropic entwickelter Standard: Systeme (z.B. Confluence, Jira, Gitlab) veröffentlichen Tools mit einem einheitlichen Interface
- Ein Tool kann sein: Lesen, Erstellen, Aktualisieren — alles was das Backend kann
- Darunter auch: **Search-Endpoints** — das RAG-Äquivalent, aber einfach als eine von mehreren Tools, und *ohne* Datenkopie

inovex

Was MCP für RAG bedeutet hat

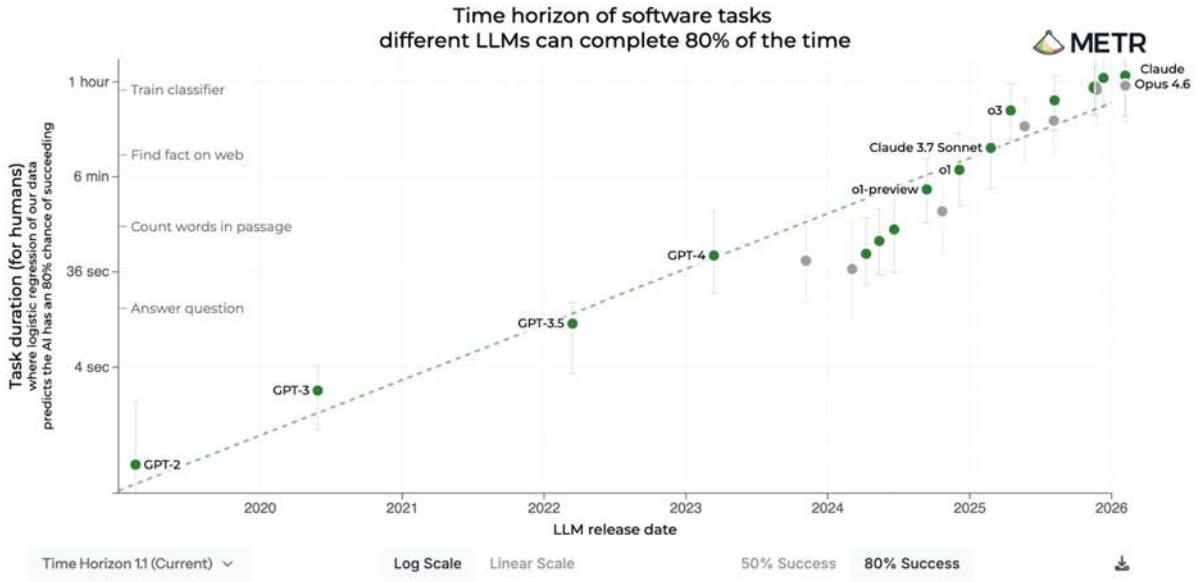
- MCP ist viel mehr als Search — aber seine Konsequenzen für die Suche waren tiefgreifend
  → Search-Endpoints für KI-Anwendungen bereitzustellen, wurde zur Aufgabe des Dateneigentümers

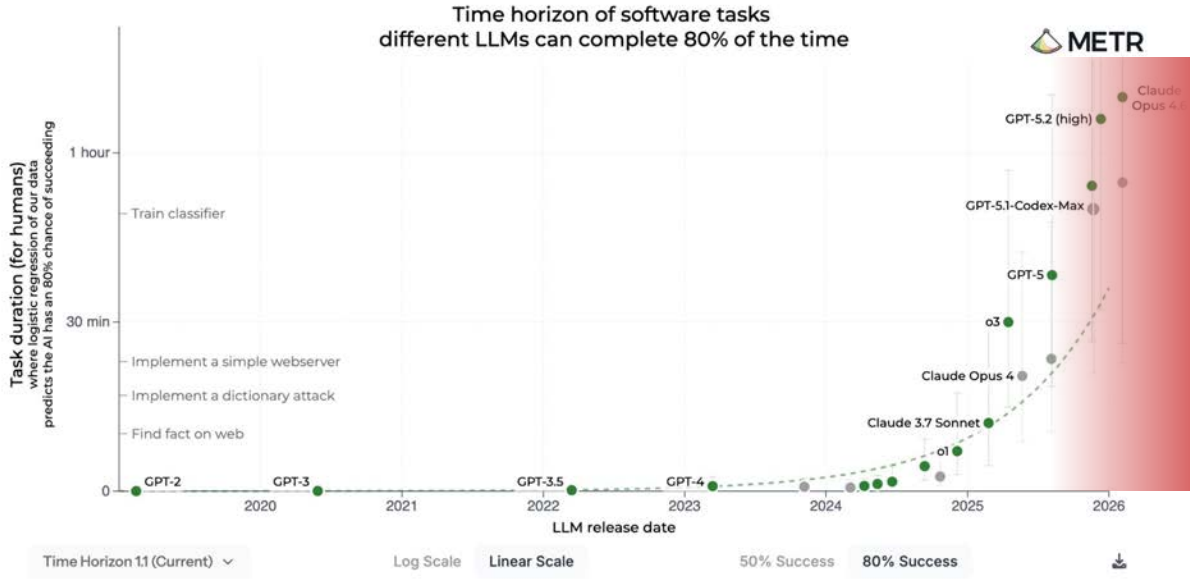
- Das LLM bringt die Intelligenz mit, auch einfache Endpoints effektiv zu nutzen
  → (vgl. Claude Code: simple Tools, aber intelligent eingesetzt)

- RAG bleibt in einigen Fällen notwendig
  — aber es ist kein Ausgangspunkt mehr

inovex

Agentic AI


Time horizon of software tasks
different LLMs can complete 80% of the time
METR
Task duration (for humans)
where logistic regression of our data
predicts the AI has an 80% chance of succeeding
1 hour
Train classifier
Find fact on web
6 min
Count words in passage
36 sec
Answer question
4 sec
GPT-2
GPT-3
GPT-3.5
GPT-4
o1-preview
o1
Claude 3.7 Sonnet
o3
Claude
Opus 4.6
2020
2021
2022
2023
2024
2025
2026
LLM release date
Time Horizon 1.1 (Current)
Log Scale
Linear Scale
50% Success
80% Success



Die exponentielle Entwicklung in den Fähigkeiten der LLMs macht es so, dass wir heute – und nicht früher – über Agentic AI sprechen

Die eigentliche Herausforderung bei Agentic AI heute

- Die LLMs sind leistungsfähig genug. Die Tools existieren.
- Die eigentliche Herausforderung bei Agentic AI: die **Steuerung** — und die **Sicherheit**
- Ein LLM allein verliert sich — Context Window füllt sich, Fokus geht verloren ("Context Rot")
- Die Frage ist nicht mehr "Was kann das LLM?", sondern "Wie steuern wir es — und wie sichern wir es ab?"

Das Agent-Harness

- Agent-Harness: ein Backend, das einem LLM ermöglicht, als fortgeschrittener Agent zu agieren
- Das Agent-Harness “steuert” das LLM:
  → Kontext-Management → Sub-Agenten → Planung → External Memory
- Beispiele: Produkte wie Claude Code, Manus, openClaw oder Frameworks wie DeepAgents (LangChain)
- Wann fertige Lösungen nutzen, wann und wie eigene bauen?
  → Sicherheit, Flexibilität, Auditability, Vorhersagbarkeit, usw.
  — das zu beantworten und zu implementieren ist unsere Aufgabe

TL;DR

Die Frage hat sich in 3.5 Jahren drastisch verändert:

2022: "Wie können LLMs Tools nutzen?"

2026: "Wie architektieren wir Systeme, in denen LLMs stundenlange autonome Tasks durchführen können ohne ein massives Security-Incident zu verursachen?"

Der Rote Faden
Warum seit 2022 Agentic AI unvermeidlich war?

- **NLP is AI-complete**: Wer Sprache beherrscht, kann “alles” (Montalvo, 1987; vgl. Collins, 2003, MIT)
- Der Moment, in dem LLMs Sprache gemeistert hatten, war der Rest nur eine Frage der Zeit
- **Sprechen** → **Handeln** → **Denken** → **Agenten**: nicht Zufall, sondern Konsequenz

inovex

Vielen Dank!

and happy networking!

inovex

Ivan Herreros Alonso
Senior AI Consultant | ML Engineer
inovex GmbH

www.linkedin.com/in/ivan-herreros-b64a204