

AI Engineering

AI Testing: ein nichtdeterministisches System testbar machen

KI-Systeme sind testbar — aber nicht mit den Oracles deterministischer Software.

Band 2 · Kapitel 33

Öffentliche Leseprobe · Auszug (§1 „Warum AI-Testing anders ist“ bis §4 „Das Test-Oracle-Problem“)

Sebahattin Özata · Diplom-Informatiker

OMEX Press

Aus Kapitel 33 in Band 2: die fünf Testschichten und das Problem, kein eindeutiges Soll-Ergebnis zu haben. Kapitelverweise beziehen sich auf das vollständige dreibändige Werk.

Kapitel 33 — AI Testing: ein nichtdeterministisches System testbar machen (fünf Schichten)

1. Warum AI-Testing anders ist

Klassisches Software-Testing kennt ein einfaches Orakel: Input → *ein* erwarteter Output; stimmt die Gleichheit, ist der Test grün. Fünf Eigenschaften generativer AI-Systeme brechen dieses Modell:

1. **Kein eindeutiger Sollwert.** Für „Fasse diesen Vertrag zusammen“ gibt es viele korrekte Antworten. Exakte Gleichheit ist das falsche Kriterium (§4).
2. **Nichtdeterminismus.** Sampling, Temperatur, Provider-Updates, Retrieval-Ranking, Tool-Antworten, Nebenläufigkeit — dieselbe Anfrage kann variieren (§3).
3. **Bewegliche Abhängigkeiten.** Modell, Prompt, Retrieval-Korpus (K32) und Policy sind *separat* versioniert; jede kann sich ändern, ohne dass sich der Code ändert.
4. **Seiteneffekte.** Ein Agent ruft Tools auf, die reale Wirkungen erzeugen — Testen darf diese Wirkungen nicht auslösen und muss sie trotzdem prüfen (§10/§11).
5. **Probabilistische Metriken.** Viele Qualitätsmaße sind selbst heuristisch oder modellbasiert (§13) — das Messinstrument ist so unsicher wie das Gemessene.

Daraus folgt nicht Resignation, sondern Methodenpflicht. Breck et al. formulierten 2017 mit dem **ML Test Score** [REAL] eine Produktionsreife-Rubrik aus 28 konkreten Tests über Daten, Modell, Infrastruktur und Monitoring — der Kernbefund: AI-Reife misst sich nicht an einer Genauigkeitszahl, sondern an der *Breite der Prüfungen*. Und Sculley et al. erinnerten schon 2015 mit **CACE** — „**Changing Anything Changes Everything**“ [REAL] daran, dass in ML-Systemen jede Änderung alles verschieben kann; genau deshalb braucht jede Änderungsart ihre eigene Regressionsprüfung (§14).

Merksatz: AI-Testing ist kein einzelner Testtyp, sondern ein *Portfolio* — und die Kunst besteht darin, für jeden Systemteil das *richtige* Orakel zu wählen.

2. Die fünf Testschichten

Das Kapitel entwickelt ein fünfstufiges Modell. Es ist keine strenge Pyramide, sondern eine Schichtung nach *Determinismusgrad* und *Signalgeschwindigkeit* — von schnell/exakt/billig unten zu langsam/probabilistisch/teuer oben.

T1 · Die fünf Testschichten

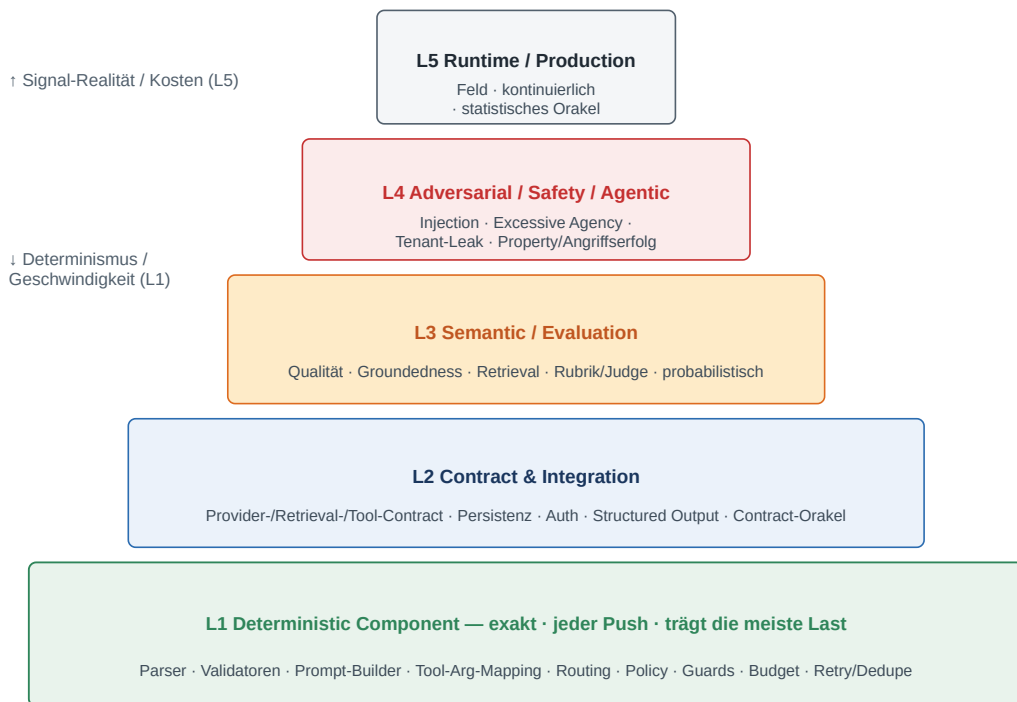
Layer	Gegenstand	Orakel	Determinismus	Geschwindigkeit
L1 Deterministic Component	Parser, Validatoren, Prompt-Builder, Tool-Arg-Mapping, Routing, Policy, Guards, Budget, Retry/Dedupe	exakt (Gleichheit/Zustand)	vollständig	schnell (CI, jeder Push)
L2 Contract & Integration	Provider-/Retrieval-/Tool-Contract, Persistenz, Auth, Feature-Flags, fail-open/closed, Structured Output	Contract/Schema	überwiegend	mittel
L3 Semantic / Evaluation	Antwortqualität, Relevanz, Groundedness, Refusal, Halluzination, Zitat-, Retrieval-, Tool-Wahl-Qualität	Rubrik/Property/Vergleich/Judge	probabilistisch	langsam
L4 Adversarial / Safety / Agentic	Injection, Poisoning, unautorisierte Tools, Excessive Agency, Tenant-Leak, Policy-Bypass, Memory-Poisoning	Property/Angriffserfolg	teils probabilistisch	mittel-langsam
L5 Runtime / Production	reale Last, Canary/Shadow, Drift, SLI/SLO (<i>Service-Level-Indikator/-Ziel</i>), Kosten-/Safety-Drift, Feedback, Incident-Signale	statistisch/Beobachtung	Feld	kontinuierlich

Die didaktische Pointe: **Die untersten Schichten tragen die meiste Last.** Die teuren, unsicheren Prüfungen (L3–L5) sind unverzichtbar, aber sie sind das *dünne obere Ende* — und sie werden wertlos, wenn die

deterministische Basis nicht steht. Wer die Autorisierung nur im teuren Semantik-Eval prüft statt als deterministische L1-Invariante, hat sie faktisch nicht geprüft.

Das fünfschichtige AI-Testing-Modell

keine strenge Pyramide — Schichtung nach Determinismusgrad und Signalgeschwindigkeit



Teste die deterministische Schale deterministisch; evaluiere den probabilistischen Kern statistisch.

Die teuren, unsicheren Prüfungen (L3–L5) werden wertlos, wenn die deterministische Basis (L1) nicht steht.

D1 · Das fünfschichtige AI-Testing-Modell

3. Determinismus — Quellen des Nichtdeterminismus

Bevor man testet, muss man wissen, *was genau* stochastisch ist.

Nichtdeterminismus kann kommen aus: Sampling, Temperatur, `top-p`, Provider-/Modell-Updates (auch stille), Retrieval-Ranking, dynamischen Daten, Tool-Antworten, zeitabhängigen Prompts, externem Zustand, Nebenläufigkeit und Agent-Schleifen.

Aber — und das ist der befreiende Punkt — **nicht jede AI-Komponente ist nichtdeterministisch**. Der Großteil eines produktiven AI-Systems ist gewöhnlicher, deterministischer Code: Wie wird der Prompt zusammengebaut? Wird das Tool-Argument korrekt gemappt? Greift der Tenant-Guard? Ist die Budget-Rechnung richtig? All das ist exakt prüfbar.

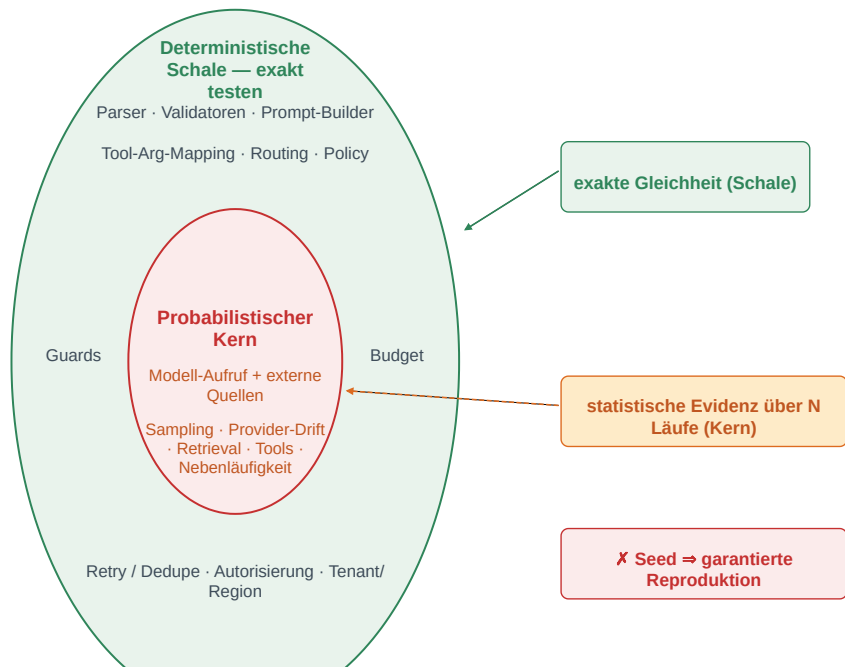
Zur **Reproduzierbarkeit** kursiert ein Missverständnis:

Ein Seed ist keine Garantie deterministischer Reproduktion. Ein *Seed* (Startwert des Zufallsgenerators) und `temperature=0` können Varianz reduzieren oder eine reproduzierbare Sampling-Sequenz ermöglichen — aber Provider-Implementierungen, Infrastruktur, Modellversionen, Fließkomma-Verhalten und Backend-Änderungen können Reproduktion trotzdem verhindern. Das ist die Testebenen-Fassung des K28-Satzes **Replay ≠ Reproduktion**: Ein gespeicherter Input erlaubt, eine Verarbeitung erneut zu *fahren*, nicht, dasselbe Ergebnis zu *erzwingen*.

Die praktische Konsequenz ist die Zerlegung in Schale und Kern (§1). Für die Schale gilt exakte Gleichheit; für den Kern gilt statistische Evidenz über viele Läufe (§14). Ein Test, der den stochastischen Kern mit exakter Gleichheit prüft, ist entweder flaky oder falsch grün.

Deterministische Schale, probabilistischer Kern

der Großteil eines AI-Systems ist gewöhnlicher, exakt testbarer Code



Nichtdeterministisch heißt nicht untestbar — es heißt: anderes Orakel.

Replay ≠ Reproduktion (K28): ein Seed reduziert Varianz, garantiert aber kein identisches Ergebnis.

D2 · Deterministische Schale, probabilistischer Kern

4. Das Test-Oracle-Problem

Der Kern jeder Teststrategie ist das **Orakel**: der Mechanismus, der entscheidet, ob ein Test bestanden hat. Barr et al. nannten 2015 in ihrer Survey [REAL] das **Test-Oracle-Problem** — die Schwierigkeit, ein solches Entscheidungsverfahren zu bekommen, wenn kein eindeutig korrekter Output existiert. Genau das ist der Normalfall bei generativer AI.

Die Antwort ist nicht „kein Orakel“, sondern *das passende* Orakel. Die folgende Sechs-Typen-Einteilung (T2) ist die **Arbeits-Taxonomie dieses Kapitels** — an Barr et al. angelehnt, aber nicht deren eigene Kategorisierung (die Survey ordnet Orakel in *spezifizierte, abgeleitete*,

implizite und *menschliche* Klassen). Sie ist auf generative AI zugeschnitten:

T2 · Test-Orakel-Typen

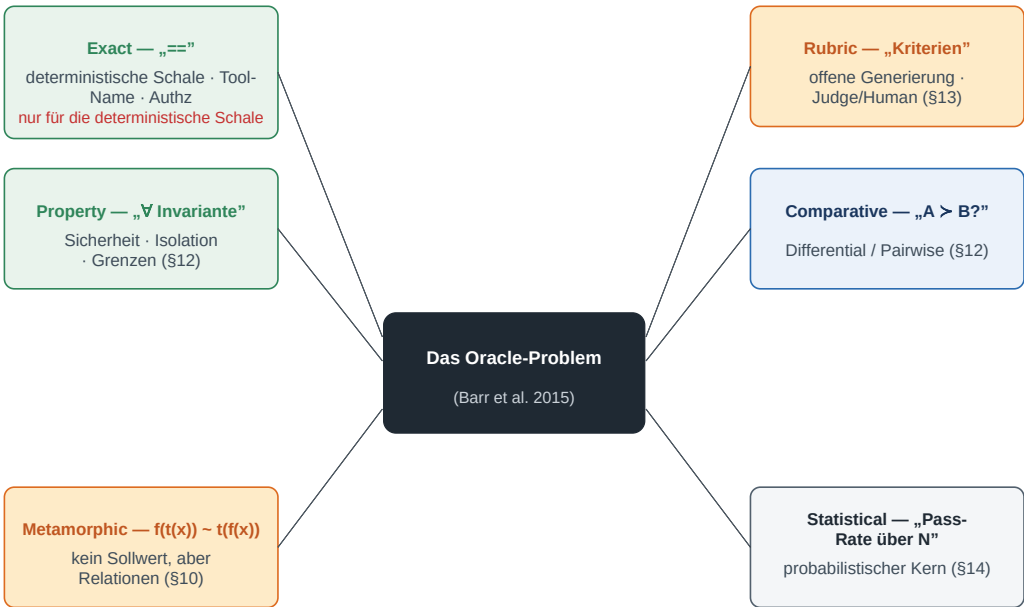
Orakel	Frage	Wann geeignet
Exact Oracle	Ist Output == erwartet?	deterministische Schale; strukturierte Felder; Tool-Name; Autorisierungs-Ausgang
Property Oracle	Gelten Invarianten immer?	Sicherheit, Isolation, Grenzen (§12) — oft stabiler als Sollantworten
Rubric Oracle	Erfüllt die Antwort definierte Kriterien?	offene Generierung; Bewertung nach Rubrik (menschlich oder Judge, §13)
Comparative Oracle	Ist A besser als B?	Modell-/Prompt-Vergleich (Differential/Pairwise, §14)
Metamorphic Oracle	Ändert sich der Output <i>wie erwartet</i> bei einer Eingabe-Transformation?	kein Sollwert bekannt, aber Relationen gelten (§12)
Statistical Oracle	Liegt die Pass-Rate/der Mittelwert über N Läufen im Rahmen?	probabilistischer Kern (§14)

Der zentrale Satz:

Die Wahl des Orakels ist die eigentliche Testentwurfs-Entscheidung. Ein falsches Orakel macht einen nützlichen Test entweder nutzlos (exakte Gleichheit auf freien Text) oder trügerisch grün (nur die Endantwort statt der Invarianten prüfen).

AI-Test-Orakel-Typen

wie entscheide ich pass/fail ohne eindeutigen Sollwert? — mit dem passenden Orakel je Prüfung



Das Oracle-Problem löst man nicht mit einem Orakel, sondern mit dem passenden je Prüfung.

Ein falsches Orakel macht den Test nutzlos (exakt auf freien Text) oder trügerisch grün (nur Endantwort statt Invarianten).

D3 · AI-Test-Orakel-Typen

Markierungen in dieser Leseprobe

Einzelne Sätze tragen eine Markierung, die sagt, woher die Aussage stammt und wie belastbar sie ist. Hier stehen nur die, die in dieser Leseprobe vorkommen.

Markierung	Bedeutung
[REAL]	ein dokumentierter realer Vorfall, kein konstruiertes Beispiel.