MCAI / MICAI - الذكاء المعرفي التركيبي الحاكمي

نظرية Moussaif Integrative Cognitive AI: من توليد الجواب إلى حاكمية شروط إنتاج الجواب

زهير مساعف - Zouhair Moussaif

طبعة بحثية موسعة: 187 صفحة - 2026

كتاب بحثي عربي كثيف يبني برنامجا نظريا وتطبيقيا وتقنيا حول Cognitive AI، ويحوّل مشروع MCAI / MICAI إلى معمارية قابلة للنقد، القياس، المحاكاة، والبرمجة.

الذكاء المعرفي ليس قدرة على إنتاج الجواب، بل قدرة على حكم شروط إنتاج الجواب.

صفحة العنوان والتنبيه المنهجي

هذا العمل طبعة بحثية موسعة تستوعب النسخ السابقة حول Cognitive AI وMoussaif Integrative Cognitive AI، ثم تضيف طبقة تطويرية هجومية: نظريات جديدة، معادلات، Benchmarks، أدوات برمجية، ومحاكاة توضيحية. لا يدّعي الكتاب أنه أثبت نظرية نهائية؛ بل يبني برنامجا بحثيا قابلا للهدم والتحسين.

مراتب الدليل
الثابت = ما له دعم واسع أو تعريف مؤسسي. الراجح = ما تؤيده أبحاث كثيرة مع بقاء فجوات. الخلافي = ما تتنازع فيه المدارس. الفرضية = تركيب جديد قابل للاختبار. التأمل = فكرة موجهة لا تُعرض كحقيقة.

الكتاب يميّز بين المصادر الخارجية: CoALA، مسوح الذاكرة، AlphaEvolve، OWASP، SWE-bench، world models، test-time scaling؛ وبين التركيب المساعفي الجديد: HAKIM، MCD، MRG، الذاكرة الوعائية، محكمة الهدم، ومقاومة الاختطاف.

```
Claim_book := StateOfArt ∪ CriticalSynthesis ∪ MoussaifModel ∪ Simulation ∪ RefutationCourt
TruthStatus(x) ∈ {ثابت، راجح، خلافي، فرضية، تركيب فلسفي، تأمل غير مثبت}
```

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

البروتوكول المعرفي - صفحة 1

الفهرس التفصيلي

الباب 1 - بروتوكول البحث وبنية الكتاب: الصفحات 1-12. يتضمن تأسيسا، تفكيكا، نمذجة، نقدا، وتطبيقات.

الباب 2 - الجذور: من المعمارية المعرفية إلى الوكيل اللغوي: الصفحات 13-30. يتضمن تأسيسا، تفكيكا، نمذجة، نقدا، وتطبيقات.

الباب 3 - الحالة الراهنة: ذاكرة، نماذج عالم، توسيع وقت الاختبار: الصفحات 31-50. يتضمن تأسيسا، تفكيكا، نمذجة، نقدا، وتطبيقات.

الباب 4 - أزمة الذكاء الحالي: الجواب بلا حاكمية: الصفحات 51-68. يتضمن تأسيسا، تفكيكا، نمذجة، نقدا، وتطبيقات.

الباب 5 - MICAI: المشروع المساعفي للذكاء المعرفي التركيبي: الصفحات 69-90. يتضمن تأسيسا، تفكيكا، نمذجة، نقدا، وتطبيقات.

الباب 6 - MRG و MCD: الصياغة الرياضية: الصفحات 91-115. يتضمن تأسيسا، تفكيكا، نمذجة، نقدا، وتطبيقات.

الباب 7 - HAKIM Architecture: التصميم التقني والبرمجي: الصفحات 116-139. يتضمن تأسيسا، تفكيكا، نمذجة، نقدا، وتطبيقات.

الباب 8 - التدريب والتقييم والمحاكاة: الصفحات 140-161. يتضمن تأسيسا، تفكيكا، نمذجة، نقدا، وتطبيقات.

الباب 9 - الأدوات التقنية غير المسبوقة وخطة المنافسة: الصفحات 162-173. يتضمن تأسيسا، تفكيكا، نمذجة، نقدا، وتطبيقات.

الباب 10 - محكمة الهدم والنسخة المحسنة: الصفحات 174-182. يتضمن تأسيسا، تفكيكا، نمذجة، نقدا، وتطبيقات.

الملاحق - المصادر، الجداول، القاموس، سجل البناء: الصفحات 183-187. يتضمن تأسيسا، تفكيكا، نمذجة، نقدا، وتطبيقات.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

البروتوكول المعرفي - صفحة 2

وكيل البحث المكتبي: يجمع المصادر المفتوحة والخرائط المرجعية ويستبعد ما لا يمكن توثيقه.

وكيل البحث الأوراق العلمية: يربط أحدث مسارات agentic AI بالمعماريات المعرفية القديمة.

وكيل البحث الأطروحات: يفتش عن الأسئلة التي لا تظهر في المقالات القصيرة: الاستمرارية، القياس، الذاكرة، العقلانية.

وكيل البحث النقد: يمنع كل مفهوم من التحول إلى شعار؛ يطلب تعريفه وتشغيله وفشله.

وكيل البحث التركيب: يحوّل التوتر بين المدارس إلى نموذج أعلى لا إلى تسوية لفظية.

وكيل البحث التحقق: يربط الادعاءات بالمصادر ويضع علامة على ما يحتاج تحققا إضافيا.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

البروتوكول المعرفي - صفحة 3

بروتوكول Anti-Summary
بعد كل مصدر نسأل: ما الذي لا يقوله؟ ما فرضيته الخفية؟ أين يقفز من قياس إلى تأويل؟ ما المتغير الغائب؟ كيف نختبره لا كيف نعيد صياغته؟

الكتاب ليس تعليقا على الأدبيات؛ الأدبيات هنا مادة خام لبناء هندسة معرفية. لذلك يستعمل المسوح الحديثة كأرضية، ثم يضيف طبقات: حاكمية، اختطاف، ذاكرة وعائية، MCD، وHAKIM-Bench.

```
Read(source) -> {claim, mechanism, blind_spot, conflict, usable_component, falsifier}
```

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

البروتوكول المعرفي - صفحة 4

مصفوفة المصادر المركزية

المصدر	نوعه	وظيفته
CoALA	ورقة/إطار معماري	ربط وكلاء اللغة بتاريخ المعماريات المعرفية: ذاكرة معيارية، أفعال داخلية/خارجية، وسياسة قرار عامة.
TTS	مسح حديث	تأسيس طبقة compute عند الاستدلال: أخذ عينات، بحث، نقد ذاتي، موسعات تفكير، ومقاييس كلفة/فائدة.
MemoryImpact	دراسة تجريبية	استخراج قانون الخبرة المتبعة وخطر انتقال الخطأ من الذاكرة؛ أساس مفهوم الذاكرة الوعائية المحكومة.
WorldModels	مسح state of art	نماذج العالم كمولدات تنبؤ ومحاكاة مضادة للواقع؛ أساس طبقة النموذج العالمي الإحالي.
AlphaEvolve	ورقة/نظام اكتشاف	إثبات أن الاكتشاف الآلي يحتاج حلقة توليد-تقييم-انتقاء لا نموذج لغة فقط؛ أساس مختبر التطور الخوارزمي في MICAI.
SWEbench	Benchmark	قياس الوكلاء البرمجيين بمهام GitHub واقعية؛ نمده إلى HAKIM-Bench لقياس الحاكمية لا النجاح فقط.
OWASP	معيار/أمن تطبيقي	تعريف المخاطر البنيوية للاختطاف النصي وغير المباشر؛ أساس Anti-Hijack Governance.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

البروتوكول المعرفي - صفحة 5

قاعدة القراءة الهجومية
لا تُقبل فكرة في MICAI إلا إذا أمكن تحويلها إلى: تعريف، متغير، آلية، مقياس، تجربة، اعتراض، ونسخة محسنة بعد النقد.

بهذا تتحول الفلسفة من زخرفة إلى هندسة. مفاهيم مثل الوعاء، الحاكمية، الاختطاف، الغاية، والزمن الداخلي لا تُترك كصور بل تُربط بعمليات: اختيار، وزن، استرجاع، تحقق، فعل، وتعلم.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

الملخص التنفيذي - 1

القضية المركزية: الجيل الحالي من النماذج يملك قدرة توليد عظيمة، لكنه لا يملك بالضرورة سيادة معرفية على شروط إنتاج الجواب. قد يجيب، يشرح، يبرمج، ويربط؛ لكنه قد يخلط المصدر بالمطلوب، الذاكرة بالحقيقة، التشابه بالسبب، والسهولة بالصواب.

أطروحة الكتاب: MCAI/MICAI يقترح نقل مركز الذكاء من output إلى governance: من ماذا قال النموذج إلى كيف حكم شروط القول. الذكاء المعرفي = ذاكرة محكومة + نموذج عالم إحالي + اعتراض داخلي + فعل قابل للتحقق + سجل إثبات.

الفرق عن RAG العادي: RAG يسترجع وثائق. MICAI يسأل: أي وعاء استرجاع؟ أي مصدر؟ ما صلاحية الذكرى؟ ما خطر انتقال الخطأ؟ هل الدليل مناسب للسؤال أم قريب منه دلاليا فقط؟

الخلاصة التنفيذية
لا نبني نموذجا لغويا أكبر؛ نبني جهازا معرفيا يجعل أي نموذج لغوي أقل قابلية للاختطاف وأكثر قدرة على إنتاج معرفة موثقة وقابلة للهدم.

```
MCAI = LLM + Memory^Ω + WorldModel^ref + Governance + Refutation + Metrics + Tools
```

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

الملخص التنفيذي - 2

أطروحة الكتاب: MCAI/MCAI يقترح نقل مركز الذكاء من output إلى governance: من ماذا قال النموذج إلى كيف حكم شروط القول. الذكاء المعرفي = ذاكرة محكومة + نموذج عالم إحالي + اعتراض داخلي + فعل قابل للتحقق + سجل إثبات.

الفرق عن RAG العادي: RAG يسترجع وثائق. MICAI يسأل: أي وعاء استرجاع؟ أي مصدر؟ ما صلاحية الذكرى؟ ما خطر انتقال الخطأ؟ هل الدليل مناسب للسؤال أم قريب منه دلاليا فقط؟

الفرق عن Agents العادية: الوكيل العادي يخطط ويستعمل أدوات. وكيل HAKIM يراقب الحاكمية: هل الأداة اختطفت الهدف؟ هل الذاكرة القديمة قادت إلى تكرار خطأ؟ هل معيار النجاح ضيق؟

الخلاصة التنفيذية
لا نبني نموذجا لغويا أكبر، نبني جهازا معرفيا يجعل أي نموذج لغوي أقل قابلية للاختطاف وأكثر قدرة على إنتاج معرفة موثقة وقابلة للهدم.

```
MCAI = LLM + Memory^Ω + WorldModel^ref + Governance + Refutation + Metrics + Tools
```

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الملخص التنفيذي - 3

الفرق عن RAG العادي: RAG يسترجع وثائق. MICAI يسأل: أي وعاء استرجاع؟ أي مصدر؟ ما صلاحية الذكرى؟ ما خطر انتقال الخطأ؟ هل الدليل مناسب للسؤال أم قريب منه دلاليا فقط؟

الفرق عن Agents العادية: الوكيل العادي يخطط ويستعمل أدوات. وكيل HAKIM يراقب الحاكمية: هل الأداة اختطفت الهدف؟ هل الذاكرة القديمة قادت إلى تكرار خطأ؟ هل معيار النجاح ضيق؟

القيمة البحثية: المشروع لا ينافس بحجم النموذج، بل بطبقة تنظيمية فوق النماذج: حاكمية، ذاكرة، اعتراض، قياس، وأمان. هذه منطقة لم تُحسم، وهي قابلة للنمذجة والتجريب.

الخلاصة التنفيذية
لا نبني نموذجا لغويا أكبر؛ نبني جهازا معرفيا يجعل أي نموذج لغوي أقل قابلية للاختطاف وأكثر قدرة على إنتاج معرفة موثقة وقابلة للهدم.

```
MCAI = LLM + Memory^Ω + WorldModel^ref + Governance + Refutation + Metrics + Tools
```

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

الملخص التنفيذي - 4

الفرق عن Agents العادية: الوكيل العادي يخطط ويستعمل أدوات. وكيل HAKIM يراقب الحاكمية: هل الأداة اختطفت الهدف؟ هل الذاكرة القديمة قادت إلى تكرار خطأ؟ هل معيار النجاح ضيق؟

القيمة البحثية: المشروع لا ينافس بحجم النموذج، بل بطبقة تنظيمية فوق النماذج: حاكمية، ذاكرة، اعتراض، قياس، وأمان. هذه منطقة لم تُحسم، وهي قابلة للنمذجة والتجريب.

القيمة الهندسية: يمكن تحويل الكتاب إلى stack: Vascular Memory Ledger، Evidence Lattice، Anti-Hijack Firewall، HAKIM Controller، Refutation Court Engine، MCD Simulator، وHAKIM-Bench.

الخلاصة التنفيذية
لا نبني نموذجا لغويا أكبر؛ نبني جهازا معرفيا يجعل أي نموذج لغوي أقل قابلية للاختطاف وأكثر قدرة على إنتاج معرفة موثقة وقابلة للهدم.

```
MCAI = LLM + Memory^Ω + WorldModel^ref + Governance + Refutation + Metrics + Tools
```

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

الملخص التنفيذي - 5

القيمة البحثية: المشروع لا ينافس بحجم النموذج، بل بطبقة تنظيمية فوق النماذج: حاكمية، ذاكرة، اعتراض، قياس، وأمان. هذه منطقة لم تُحسم، وهي قابلة للنمذجة والتجريب.

القيمة الهندسية: يمكن تحويل الكتاب إلى stack: Vascular Memory Ledger، Evidence Lattice، Anti-Hijack Firewall، HAKIM Controller، Refutation Court Engine، MCD Simulator، وHAKIM-Bench.

الخلاصة التنفيذية
لا نبني نموذجا لغويا أكبر؛ نبني جهازا معرفيا يجعل أي نموذج لغوي أقل قابلية للاختطاف وأكثر قدرة على إنتاج معرفة موثقة وقابلة للهدم.

```
MCAI = LLM + Memory^Ω + WorldModel^ref + Governance + Refutation + Metrics + Tools
```

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

سجل المصادر التحليلي - صفحة 1

CoALA - Cognitive Architectures for Language Agents

النوع والسنة: ورقة/إطار معماري - 2023/2024. **المؤلف/المؤسسة:** Sumers, Yao, Narasimhan, Griffiths.

ما أضافه للكتاب: ربط وكلاء اللغة بتاريخ المعماريات المعرفية: ذاكرة معيارية، أفعال داخلية/خارجية، وسياسة قرار عامة.

حدوده: لا يستعمل هنا كسلطة مطلقة؛ بل كجزء من مصفوفة. كل مصدر يثبت زاوية ويترك فجوة: الذاكرة بلا حاكمية، القياس بلا معنى، أو الأمان بلا نظرية معرفية.

TTS - A Survey on Test-Time Scaling in Large Language Models

النوع والسنة: مسح حديث - 2025. **المؤلف/المؤسسة:** Zhang et al..

ما أضافه للكتاب: تأسيس طبقة compute عند الاستدلال: أخذ عينات، بحث، نقد ذاتي، موسعات تفكير، ومقاييس كلفة/فائدة.

حدوده: لا يستعمل هنا كسلطة مطلقة؛ بل كجزء من مصفوفة. كل مصدر يثبت زاوية ويترك فجوة: الذاكرة بلا حاكمية، القياس بلا معنى، أو الأمان بلا نظرية معرفية.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

سجل المصادر التحليلي - صفحة 2

MemoryImpact - How Memory Management Impacts LLM Agents

النوع والسنة: دراسة تجريبية - 2025. **المؤلف/المؤسسة:** Xiong et al..

ما أضافه للكتاب: استخراج قانون الخبرة المتبعة وخطر انتقال الخطأ من الذاكرة؛ أساس مفهوم الذاكرة الوعائية المحكومة.

حدوده: لا يستعمل هنا كسلطة مطلقة؛ بل كجزء من مصفوفة. كل مصدر يثبت زاوية ويترك فجوة: الذاكرة بلا حاكمية، القياس بلا معنى، أو الأمان بلا نظرية معرفية.

WorldModels - A Comprehensive Survey on World Models for Embodied AI

النوع والسنة: مسح state of art - 2025. **المؤلف/المؤسسة:** Li et al..

ما أضافه للكتاب: نماذج العالم كمولدات تنبؤ ومحاكاة مضادة للواقع؛ أساس طبقة النموذج العالمي الإحالي.

حدوده: لا يستعمل هنا كسلطة مطلقة؛ بل كجزء من مصفوفة. كل مصدر يثبت زاوية ويترك فجوة: الذاكرة بلا حاكمية، القياس بلا معنى، أو الأمان بلا نظرية معرفية.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

سجل المصادر التحليلي - صفحة 3

AlphaEvolve - AlphaEvolve: A coding agent for scientific and algorithmic discovery

النوع والسنة: ورقة/نظام اكتشاف - 2025. **المؤلف/المؤسسة:** Novikov et al. / Google DeepMind.

ما أضافه للكتاب: إثبات أن الاكتشاف الآلي يحتاج حلقة توليد-تقييم-انتقاء لا نموذج لغة فقط؛ أساس مختبر التطور الخوارزمي في MICAI.

حدوده: لا يستعمل هنا كسلطة مطلقة؛ بل كجزء من مصفوفة. كل مصدر يثبت زاوية ويترك فجوة: الذاكرة بلا حاكمية، القياس بلا معنى، أو الأمان بلا نظرية معرفية.

SWEbench - SWE-bench: Can Language Models Resolve Real-World GitHub Issues?

النوع والسنة: Benchmark - 2024. **المؤلف/المؤسسة:** Jimenez et al..

ما أضافه للكتاب: قياس الوكلاء البرمجيين بمهام GitHub واقعية؛ نمده إلى HAKIM-Bench لقياس الحاكمية لا النجاح فقط.

حدوده: لا يستعمل هنا كسلطة مطلقة؛ بل كجزء من مصفوفة. كل مصدر يثبت زاوية ويترك فجوة: الذاكرة بلا حاكمية، القياس بلا معنى، أو الأمان بلا نظرية معرفية.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

سجل المصادر التحليلي - صفحة 4

OWASP - LLM01:2025 Prompt Injection

النوع والسنة: معيار/أمن تطبيقي - 2025. **المؤلف/المؤسسة:** OWASP GenAI Security Project.

ما أضافه للكتاب: تعريف المخاطر البنيوية للاختطاف النصي وغير المباشر؛ أساس Anti-Hijack Governance.

حدوده: لا يستعمل هنا كسلطة مطلقة؛ بل كجزء من مصفوفة. كل مصدر يثبت زاوية ويترك فجوة: الذاكرة بلا حاكمية، القياس بلا معنى، أو الأمان بلا نظرية معرفية.

ACTR - ACT-R: A cognitive architecture tutorial

النوع والسنة: معمارية معرفية - تاريخي/تأسيسي. **المؤلف/المؤسسة:** Anderson/ACT-R community.

ما أضافه للكتاب: الفصل بين البنية الثابتة والمعرفة الخاصة بالمهمة، وتحويل العقل إلى محاكاة قابلة للتنبؤ.

حدوده: لا يستعمل هنا كسلطة مطلقة؛ بل كجزء من مصفوفة. كل مصدر يثبت زاوية ويترك فجوة: الذاكرة بلا حاكمية، القياس بلا معنى، أو الأمان بلا نظرية معرفية.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

سجل المصادر التحليلي - صفحة 5

Soar - Soar Cognitive Architecture

النوع والسنة: معمارية معرفية - تاريخي/تأسيسي. **المؤلف/المؤسسة:** Laird/Newell/Soar community.

ما أضافه للكتاب: القرار الدوري، الإنتاجات، التعلم بالتحزيم، والوكيل العام القابل لمجالات متعددة.

حدوده: لا يستعمل هنا كسلطة مطلقة؛ بل كجزء من مصفوفة. كل مصدر يثبت زاوية ويترك فجوة: الذاكرة بلا حاكمية، القياس بلا معنى، أو الأمان بلا نظرية معرفية.

LIDA - LIDA: systems-level architecture for cognition, emotion and learning

النوع والسنة: معمارية معرفية - تاريخي/تأسيسي. **المؤلف/المؤسسة:** Franklin et al..

ما أضافه للكتاب: دورة الإدراك-الفعل والانتباه والوعي الوظيفي كآلية اختيار عالمية.

حدوده: لا يستعمل هنا كسلطة مطلقة؛ بل كجزء من مصفوفة. كل مصدر يثبت زاوية ويترك فجوة: الذاكرة بلا حاكمية، القياس بلا معنى، أو الأمان بلا نظرية معرفية.

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

سجل المصادر التحليلي - صفحة 6

FreeEnergy - Active Inference / Free Energy Principle

النوع والسنة: نظرية معرفية-حيوية - تأسيسي/مستمر. **المؤلف/المؤسسة:** Friston et al..

ما أضافه للكتاب: التنبؤ، تقليل المفاجأة، الاختيار الفعال، تُستخدم هنا كقرين لا كاختزال للوعي.

حدوده: لا يستعمل هنا كسلطة مطلقة، بل كجزء من مصفوفة. كل مصدر يثبت زاوية ويترك فجوة: الذاكرة بلا حاكمية، القياس بلا معنى، أو الأمان بلا نظرية معرفية.

IITGNW - Global Workspace, IIT, Predictive Processing

النوع والسنة: علوم وعي - تأسيسي/جدلي. **المؤلف/المؤسسة:** Baars, Dehaene, Tononi, Clark....

ما أضافه للكتاب: تمييز الوعي كإتاحة عالمية أو تكامل معلوماتي أو تنبؤ جسدي، تُستعمل كعدسات لا كحسم.

حدوده: لا يستعمل هنا كسلطة مطلقة، بل كجزء من مصفوفة. كل مصدر يثبت زاوية ويترك فجوة: الذاكرة بلا حاكمية، القياس بلا معنى، أو الأمان بلا نظرية معرفية.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

سجل المصادر التحليلي - صفحة 7

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

سجل المصادر التحليلي - صفحة 8

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

سجل المصادر التحليلي - صفحة 9

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

سجل المصادر التحليلي - صفحة 10

ملاحظة تحقق
بعض المصادر أحدث من الكتب الجامعية وتبقى preprint، لذلك وُضعت في مرتبة راجح/حديث لا ثابت نهائي. الاعتماد عليها في كشف الاتجاه لا في إعلان الإجماع.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الجذور والتأسيس - من المعمارية المعرفية إلى الوكيل

المعماريات المعرفية التاريخية حاولت أن تحدد ما يبقى ثابتا عبر المهام: الذاكرة، آليات القرار، التمثيل، التعلم. وكلاء اللغة أعادوا السؤال لكن بوسيط جديد: نموذج لغوي واسع يصبح قلبا احتماليا داخل بنية أفعال وذاكرة.

النتيجة التي نستخرجها: كل محاولة لبناء AI معرفي يجب أن تنتقل من وصف المكونات إلى تعريف علاقات السلطة بينها. من يقرر ما يُسترجع؟ من يصدق الدليل؟ من يوقف الأداة؟ من يغير الخطة؟ هذه أسئلة حاكمية.

تمييز حاسم
Cognitive AI ليس LLM + أدوات فقط. هو نظام يملك ذاكرة ذات صلاحية، نموذجا للعالم، محكمة اعتراض، وميزانا لتكلفة التفكير والفعل.

الخطأ الشائع هنا هو اعتبار الزيادة في الحجم بديلا عن البناء المعماري. الحجم يعطي مادة احتمالية؛ الحاكمية تجعلها معرفة قابلة للمساءلة.

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

الجذور والتأسيس - ACT-R/Soar/LIDA كجذور

ACT-R يعلّمنا أن البنية والمعرفة منفصلتان، Soar يعلّمنا أن القرار الدوري والتعلم من التعثر قاعدة عامة، LIDA يعلّمنا أن الانتباه ودورة الفعل ليست إضافات بل مركز تنظيم السلوك.

النتيجة التي نستخرجها: كل محاولة لبناء AI معرفي يجب أن تنتقل من وصف المكونات إلى تعريف علاقات السلطة بينها. من يقرر ما يُسترجع؟ من يصدق الدليل؟ من يوقف الأداة؟ من يغير الخطة؟ هذه أسئلة حاكمية.

تمييز حاسم
Cognitive AI ليس LLM + أدوات فقط. هو نظام يملك ذاكرة ذات صلاحية، نموذجا للعالم، محكمة اعتراض، وميزانا لتكلفة التفكير والفعل.

```
Agent_t = <State, Memory, Policy, Tools, World, Goals, Governance>
Governance := Select(what_to_trust, what_to_do, when_to_stop, how_to_refute)
```

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الجذور والتأسيس - CoALA كجسر

CoALA مهم لأنه لا يبدأ من فراغ: يترجم تاريخ الإنتاجات والذاكرة والأفعال إلى لغة وكلاء LLM. لكن الكتاب يرى أن CoALA يحتاج طبقة حاكمية إحالية تقيس الاختطاف وشروط الفشل.

النتيجة التي نستخرجها: كل محاولة لبناء AI معرفي يجب أن تنتقل من وصف المكونات إلى تعريف علاقات السلطة بينها. من يقرر ما يُسترجع؟ من يصدق الدليل؟ من يوقف الأداة؟ من يغير الخطة؟ هذه أسئلة حاكمية.

تمييز حاسم
Cognitive AI ليس LLM + أدوات فقط. هو نظام يملك ذاكرة ذات صلاحية، نموذجا للعالم، محكمة اعتراض، وميزانا لتكلفة التفكير والفعل.

الخطأ الشائع هنا هو اعتبار الزيادة في الحجم بديلا عن البناء المعماري. الحجم يعطي مادة احتمالية؛ الحاكمية تجعلها معرفة قابلة للمساءلة.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

الجذور والتأسيس - التناقض البنيوي

النموذج اللغوي واسع جدا لكنه ليس حاكما بذاته. الأدوات قوية لكنها لا تعرف الغاية. الذاكرة مفيدة لكنها تنقل الخطأ. الاختبار يعطي رقما لكنه قد يخفي سوء الفهم. من هنا يولد MICAI.

النتيجة التي نستخرجها: كل محاولة لبناء AI معرفي يجب أن تنتقل من وصف المكونات إلى تعريف علاقات السلطة بينها. من يقرر ما يُسترجع؟ من يصدق الدليل؟ من يوقف الأداة؟ من يغير الخطة؟ هذه أسئلة حاكمية.

تمييز حاسم
Cognitive AI ليس LLM + أدوات فقط. هو نظام يملك ذاكرة ذات صلاحية، نموذجا للعالم، محكمة اعتراض، وميزانا لتكلفة التفكير والفعل.

الخطأ الشائع هنا هو اعتبار الزيادة في الحجم بديلا عن البناء المعماري. الحجم يعطي مادة احتمالية؛ الحاكمية تجعلها معرفة قابلة للمساءلة.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

الجذور والتأسيس - التأسيس الفلسفي

النموذج ليس الواقع. الدليل ليس الذاكرة. القرب الدلالي ليس السببية. النجاح في benchmark ليس فهما عاما. هذه التمييزات لا تزينية؛ هي شروط أمان معرفي.

النتيجة التي نستخرجها: كل محاولة لبناء AI معرفي يجب أن تنتقل من وصف المكونات إلى تعريف علاقات السلطة بينها. من يقرر ما يُسترجع؟ من يصدق الدليل؟ من يوقف الأداة؟ من يغير الخطة؟ هذه أسئلة حاكمية.

تمييز حاسم
Cognitive AI ليس LLM + أدوات فقط. هو نظام يملك ذاكرة ذات صلاحية، نموذجا للعالم، محكمة اعتراض، وميزانا لتكلفة التفكير والفعل.

الخطأ الشائع هنا هو اعتبار الزيادة في الحجم بديلا عن البناء المعماري. الحجم يعطي مادة احتمالية؛ الحاكمية تجعلها معرفة قابلة للمساءلة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

الجذور والتأسيس - النظم المعقدة

الوكيل ليس مكونا مفردا بل نظام حلقات: إدراك، ذاكرة، تفسير، فعل، تقييم، تعلم. لذلك يظهر الانبثاق والانحراف والتراكم الخاطئ. الحاكمية هي ضبط هذه الحلقات.

النتيجة التي نستخرجها: كل محاولة لبناء AI معرفي يجب أن تنتقل من وصف المكونات إلى تعريف علاقات السلطة بينها. من يقرر ما يُسترجع؟ من يصدق الدليل؟ من يوقف الأداة؟ من يغير الخطة؟ هذه أسئلة حاكمية.

تمييز حاسم
Cognitive AI ليس LLM + أدوات فقط. هو نظام يملك ذاكرة ذات صلاحية، نموذجا للعالم، محكمة اعتراض، وميزانا لتكلفة التفكير والفعل.

```
Agent_t = <State, Memory, Policy, Tools, World, Goals, Governance>
Governance := Select(what_to_trust, what_to_do, when_to_stop, how_to_refute)
```

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الجذور والتأسيس - نظرية المعرفة

العلم يتقدم حين يميز بين مراتب اليقين. MICAI يفرض على الوكيل أن يصنف ادعاءه: ثابت، راجح، خلافي، فرضية، أو تركيب. هذا التصنيف جزء من الجواب لا ملحق أخلاقي.

النتيجة التي نستخرجها: كل محاولة لبناء AI معرفي يجب أن تنتقل من وصف المكونات إلى تعريف علاقات السلطة بينها. من يقرر ما يُسترجع؟ من يصدق الدليل؟ من يوقف الأداة؟ من يغير الخطة؟ هذه أسئلة حاكمية.

تمييز حاسم
Cognitive AI ليس LLM + أدوات فقط. هو نظام يملك ذاكرة ذات صلاحية، نموذجا للعالم، محكمة اعتراض، وميزانا لتكلفة التفكير والفعل.

الخطأ الشائع هنا هو اعتبار الزيادة في الحجم بديلا عن البناء المعماري. الحجم يعطي مادة احتمالية؛ الحاكمية تجعلها معرفة قابلة للمساءلة.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الجذور والتأسيس - الوعي كعدسة لا كادعاء

الكتاب يستعمل نظرية التراكب الوعائي للوعي كعدسة توليدية: كيف تتراكب الأوعية؟ كيف يصعد معنى؟ كيف تحكم الغاية؟ لكنه لا يدعي أن الآلة واعية ظاهراتيا.

النتيجة التي نستخرجها: كل محاولة لبناء AI معرفي يجب أن تنتقل من وصف المكونات إلى تعريف علاقات السلطة بينها. من يقرر ما يُسترجع؟ من يصدق الدليل؟ من يوقف الأداة؟ من يغير الخطة؟ هذه أسئلة حاكمية.

تمييز حاسم
Cognitive AI ليس LLM + أدوات فقط. هو نظام يملك ذاكرة ذات صلاحية، نموذجا للعالم، محكمة اعتراض، وميزانا لتكلفة التفكير والفعل.

الخطأ الشائع هنا هو اعتبار الزيادة في الحجم بديلا عن البناء المعماري. الحجم يعطي مادة احتمالية؛ الحاكمية تجعلها معرفة قابلة للمساءلة.

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

الجذور والتأسيس - من المعمارية المعرفية إلى الوكيل

المعماريات المعرفية التاريخية حاولت أن تحدد ما يبقى ثابتا عبر المهام: الذاكرة، آليات القرار، التمثيل، التعلم. وكلاء اللغة أعادوا السؤال لكن بوسيط جديد: نموذج لغوي واسع يصبح قلبا احتماليا داخل بنية أفعال وذاكرة.

النتيجة التي نستخرجها: كل محاولة لبناء AI معرفي يجب أن تنتقل من وصف المكونات إلى تعريف علاقات السلطة بينها. من يقرر ما يُسترجع؟ من يصدق الدليل؟ من يوقف الأداة؟ من يغير الخطة؟ هذه أسئلة حاكمية.

تمييز حاسم
Cognitive AI ليس LLM + أدوات فقط. هو نظام يملك ذاكرة ذات صلاحية، نموذجا للعالم، محكمة اعتراض، وميزانا لتكلفة التفكير والفعل.

الخطأ الشائع هنا هو اعتبار الزيادة في الحجم بديلا عن البناء المعماري. الحجم يعطي مادة احتمالية؛ الحاكمية تجعلها معرفة قابلة للمساءلة.

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

الجذور والتأسيس - ACT-R/Soar/LIDA كجذور

ACT-R يعلّمنا أن البنية والمعرفة منفصلتان، Soar يعلّمنا أن القرار الدوري والتعلم من التعثر قاعدة عامة، LIDA يعلّمنا أن الانتباه ودورة الفعل ليست إضافات بل مركز تنظيم السلوك.

النتيجة التي نستخرجها: كل محاولة لبناء AI معرفي يجب أن تنتقل من وصف المكونات إلى تعريف علاقات السلطة بينها. من يقرر ما يُسترجع؟ من يصدق الدليل؟ من يوقف الأداة؟ من يغير الخطة؟ هذه أسئلة حاكمية.

تمييز حاسم
Cognitive AI ليس LLM + أدوات فقط. هو نظام يملك ذاكرة ذات صلاحية، نموذجا للعالم، محكمة اعتراض، وميزانا لتكلفة التفكير والفعل.

```
Agent_t = <State, Memory, Policy, Tools, World, Goals, Governance>
Governance := Select(what_to_trust, what_to_do, when_to_stop, how_to_refute)
```

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الجذور والتأسيس - CoALA كجسر

CoALA مهم لأنه لا يبدأ من فراغ: يترجم تاريخ الإنتاجات والذاكرة والأفعال إلى لغة وكلاء LLM. لكن الكتاب يرى أن CoALA يحتاج طبقة حاكمية إحالية تقيس الاختطاف وشروط الفشل.

النتيجة التي نستخرجها: كل محاولة لبناء AI معرفي يجب أن تنتقل من وصف المكونات إلى تعريف علاقات السلطة بينها. من يقرر ما يُسترجع؟ من يصدق الدليل؟ من يوقف الأداة؟ من يغير الخطة؟ هذه أسئلة حاكمية.

تمييز حاسم
Cognitive AI ليس LLM + أدوات فقط. هو نظام يملك ذاكرة ذات صلاحية، نموذجا للعالم، محكمة اعتراض، وميزانا لتكلفة التفكير والفعل.

الخطأ الشائع هنا هو اعتبار الزيادة في الحجم بديلا عن البناء المعماري. الحجم يعطي مادة احتمالية؛ الحاكمية تجعلها معرفة قابلة للمساءلة.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

الجذور والتأسيس - التناقض البنيوي

النموذج اللغوي واسع جدا لكنه ليس حاكما بذاته. الأدوات قوية لكنها لا تعرف الغاية. الذاكرة مفيدة لكنها تنقل الخطأ. الاختبار يعطي رقما لكنه قد يخفي سوء الفهم. من هنا يولد MICAI.

النتيجة التي نستخرجها: كل محاولة لبناء AI معرفي يجب أن تنتقل من وصف المكونات إلى تعريف علاقات السلطة بينها. من يقرر ما يُسترجع؟ من يصدق الدليل؟ من يوقف الأداة؟ من يغير الخطة؟ هذه أسئلة حاكمية.

تمييز حاسم
Cognitive AI ليس LLM + أدوات فقط. هو نظام يملك ذاكرة ذات صلاحية، نموذجا للعالم، محكمة اعتراض، وميزانا لتكلفة التفكير والفعل.

الخطأ الشائع هنا هو اعتبار الزيادة في الحجم بديلا عن البناء المعماري. الحجم يعطي مادة احتمالية؛ الحاكمية تجعلها معرفة قابلة للمساءلة.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

الجذور والتأسيس - التأسيس الفلسفي

النموذج ليس الواقع. الدليل ليس الذاكرة. القرب الدلالي ليس السببية. النجاح في benchmark ليس فهما عاما. هذه التمييزات لا تزينية؛ هي شروط أمان معرفي.

النتيجة التي نستخرجها: كل محاولة لبناء AI معرفي يجب أن تنتقل من وصف المكونات إلى تعريف علاقات السلطة بينها. من يقرر ما يُسترجع؟ من يصدق الدليل؟ من يوقف الأداة؟ من يغير الخطة؟ هذه أسئلة حاكمية.

تمييز حاسم
Cognitive AI ليس LLM + أدوات فقط. هو نظام يملك ذاكرة ذات صلاحية، نموذجا للعالم، محكمة اعتراض، وميزانا لتكلفة التفكير والفعل.

الخطأ الشائع هنا هو اعتبار الزيادة في الحجم بديلا عن البناء المعماري. الحجم يعطي مادة احتمالية؛ الحاكمية تجعلها معرفة قابلة للمساءلة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

الجذور والتأسيس - النظم المعقدة

الوكيل ليس مكونا مفردا بل نظام حلقات: إدراك، ذاكرة، تفسير، فعل، تقييم، تعلم. لذلك يظهر الانبثاق والانحراف والتراكم الخاطئ. الحاكمية هي ضبط هذه الحلقات.

النتيجة التي نستخرجها: كل محاولة لبناء AI معرفي يجب أن تنتقل من وصف المكونات إلى تعريف علاقات السلطة بينها. من يقرر ما يُسترجع؟ من يصدق الدليل؟ من يوقف الأداة؟ من يغير الخطة؟ هذه أسئلة حاكمية.

تمييز حاسم
Cognitive AI ليس LLM + أدوات فقط. هو نظام يملك ذاكرة ذات صلاحية، نموذجا للعالم، محكمة اعتراض، وميزانا لتكلفة التفكير والفعل.

```
Agent_t = <State, Memory, Policy, Tools, World, Goals, Governance>
Governance := Select(what_to_trust, what_to_do, when_to_stop, how_to_refute)
```

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الجذور والتأسيس - نظرية المعرفة

العلم يتقدم حين يميز بين مراتب اليقين. MICAI يفرض على الوكيل أن يصنف ادعاءه: ثابت، راجح، خلافي، فرضية، أو تركيب. هذا التصنيف جزء من الجواب لا ملحق أخلاقي.

النتيجة التي نستخرجها: كل محاولة لبناء AI معرفي يجب أن تنتقل من وصف المكونات إلى تعريف علاقات السلطة بينها. من يقرر ما يُسترجع؟ من يصدق الدليل؟ من يوقف الأداة؟ من يغير الخطة؟ هذه أسئلة حاكمية.

تمييز حاسم
Cognitive AI ليس LLM + أدوات فقط. هو نظام يملك ذاكرة ذات صلاحية، نموذجا للعالم، محكمة اعتراض، وميزانا لتكلفة التفكير والفعل.

الخطأ الشائع هنا هو اعتبار الزيادة في الحجم بديلا عن البناء المعماري. الحجم يعطي مادة احتمالية؛ الحاكمية تجعلها معرفة قابلة للمساءلة.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الجذور والتأسيس - الوعي كعدسة لا كادعاء

الكتاب يستعمل نظرية التراكب الوعائي للوعي كعدسة توليدية: كيف تتراكب الأوعية؟ كيف يصعد معنى؟ كيف تحكم الغاية؟ لكنه لا يدعي أن الآلة واعية ظاهراتيا.

النتيجة التي نستخرجها: كل محاولة لبناء AI معرفي يجب أن تنتقل من وصف المكونات إلى تعريف علاقات السلطة بينها. من يقرر ما يُسترجع؟ من يصدق الدليل؟ من يوقف الأداة؟ من يغير الخطة؟ هذه أسئلة حاكمية.

تمييز حاسم
Cognitive AI ليس LLM + أدوات فقط. هو نظام يملك ذاكرة ذات صلاحية، نموذجا للعالم، محكمة اعتراض، وميزانا لتكلفة التفكير والفعل.

الخطأ الشائع هنا هو اعتبار الزيادة في الحجم بديلا عن البناء المعماري. الحجم يعطي مادة احتمالية؛ الحاكمية تجعلها معرفة قابلة للمساءلة.

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

الجذور والتأسيس - من المعمارية المعرفية إلى الوكيل

المعماريات المعرفية التاريخية حاولت أن تحدد ما يبقى ثابتا عبر المهام: الذاكرة، آليات القرار، التمثيل، التعلم. وكلاء اللغة أعادوا السؤال لكن بوسيط جديد: نموذج لغوي واسع يصبح قلبا احتماليا داخل بنية أفعال وذاكرة.

النتيجة التي نستخرجها: كل محاولة لبناء AI معرفي يجب أن تنتقل من وصف المكونات إلى تعريف علاقات السلطة بينها. من يقرر ما يُسترجع؟ من يصدق الدليل؟ من يوقف الأداة؟ من يغير الخطة؟ هذه أسئلة حاكمية.

تمييز حاسم
Cognitive AI ليس LLM + أدوات فقط. هو نظام يملك ذاكرة ذات صلاحية، نموذجا للعالم، محكمة اعتراض، وميزانا لتكلفة التفكير والفعل.

الخطأ الشائع هنا هو اعتبار الزيادة في الحجم بديلا عن البناء المعماري. الحجم يعطي مادة احتمالية؛ الحاكمية تجعلها معرفة قابلة للمساءلة.

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

الجذور والتأسيس - ACT-R/Soar/LIDA كجذور

ACT-R يعلّمنا أن البنية والمعرفة منفصلتان، Soar يعلّمنا أن القرار الدوري والتعلم من التعثر قاعدة عامة، LIDA يعلّمنا أن الانتباه ودورة الفعل ليست إضافات بل مركز تنظيم السلوك.

النتيجة التي نستخرجها: كل محاولة لبناء AI معرفي يجب أن تنتقل من وصف المكونات إلى تعريف علاقات السلطة بينها. من يقرر ما يُسترجع؟ من يصدق الدليل؟ من يوقف الأداة؟ من يغير الخطة؟ هذه أسئلة حاكمية.

تمييز حاسم
Cognitive AI ليس LLM + أدوات فقط. هو نظام يملك ذاكرة ذات صلاحية، نموذجا للعالم، محكمة اعتراض، وميزانا لتكلفة التفكير والفعل.

```
Agent_t = <State, Memory, Policy, Tools, World, Goals, Governance>
Governance := Select(what_to_trust, what_to_do, when_to_stop, how_to_refute)
```

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الجذور والتأسيس - CoALA كجسر

CoALA مهم لأنه لا يبدأ من فراغ: يترجم تاريخ الإنتاجات والذاكرة والأفعال إلى لغة وكلاء LLM. لكن الكتاب يرى أن CoALA يحتاج طبقة حاكمية إحالية تقيس الاختطاف وشروط الفشل.

النتيجة التي نستخرجها: كل محاولة لبناء AI معرفي يجب أن تنتقل من وصف المكونات إلى تعريف علاقات السلطة بينها. من يقرر ما يُسترجع؟ من يصدق الدليل؟ من يوقف الأداة؟ من يغير الخطة؟ هذه أسئلة حاكمية.

تمييز حاسم
Cognitive AI ليس LLM + أدوات فقط. هو نظام يملك ذاكرة ذات صلاحية، نموذجا للعالم، محكمة اعتراض، وميزانا لتكلفة التفكير والفعل.

الخطأ الشائع هنا هو اعتبار الزيادة في الحجم بديلا عن البناء المعماري. الحجم يعطي مادة احتمالية؛ الحاكمية تجعلها معرفة قابلة للمساءلة.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

الجذور والتأسيس - التناقض البنيوي

النموذج اللغوي واسع جدا لكنه ليس حاكما بذاته. الأدوات قوية لكنها لا تعرف الغاية. الذاكرة مفيدة لكنها تنقل الخطأ. الاختبار يعطي رقما لكنه قد يخفي سوء الفهم. من هنا يولد MICAI.

النتيجة التي نستخرجها: كل محاولة لبناء AI معرفي يجب أن تنتقل من وصف المكونات إلى تعريف علاقات السلطة بينها. من يقرر ما يُسترجع؟ من يصدق الدليل؟ من يوقف الأداة؟ من يغير الخطة؟ هذه أسئلة حاكمية.

تمييز حاسم
Cognitive AI ليس LLM + أدوات فقط. هو نظام يملك ذاكرة ذات صلاحية، نموذجا للعالم، محكمة اعتراض، وميزانا لتكلفة التفكير والفعل.

الخطأ الشائع هنا هو اعتبار الزيادة في الحجم بديلا عن البناء المعماري. الحجم يعطي مادة احتمالية؛ الحاكمية تجعلها معرفة قابلة للمساءلة.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

الجذور والتأسيس - التأسيس الفلسفي

النموذج ليس الواقع. الدليل ليس الذاكرة. القرب الدلالي ليس السببية. النجاح في benchmark ليس فهما عاما. هذه التمييزات لا تزينية؛ هي شروط أمان معرفي.

النتيجة التي نستخرجها: كل محاولة لبناء AI معرفي يجب أن تنتقل من وصف المكونات إلى تعريف علاقات السلطة بينها. من يقرر ما يُسترجع؟ من يصدق الدليل؟ من يوقف الأداة؟ من يغير الخطة؟ هذه أسئلة حاكمية.

تمييز حاسم
Cognitive AI ليس LLM + أدوات فقط. هو نظام يملك ذاكرة ذات صلاحية، نموذجا للعالم، محكمة اعتراض، وميزانا لتكلفة التفكير والفعل.

الخطأ الشائع هنا هو اعتبار الزيادة في الحجم بديلا عن البناء المعماري. الحجم يعطي مادة احتمالية؛ الحاكمية تجعلها معرفة قابلة للمساءلة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

الجذور والتأسيس - النظم المعقدة

الوكيل ليس مكونا مفردا بل نظام حلقات: إدراك، ذاكرة، تفسير، فعل، تقييم، تعلم. لذلك يظهر الانبثاق والانحراف والتراكم الخاطئ. الحاكمية هي ضبط هذه الحلقات.

النتيجة التي نستخرجها: كل محاولة لبناء AI معرفي يجب أن تنتقل من وصف المكونات إلى تعريف علاقات السلطة بينها. من يقرر ما يُسترجع؟ من يصدق الدليل؟ من يوقف الأداة؟ من يغير الخطة؟ هذه أسئلة حاكمية.

تمييز حاسم
Cognitive AI ليس LLM + أدوات فقط. هو نظام يملك ذاكرة ذات صلاحية، نموذجا للعالم، محكمة اعتراض، وميزانا لتكلفة التفكير والفعل.

```
Agent_t = <State, Memory, Policy, Tools, World, Goals, Governance>
Governance := Select(what_to_trust, what_to_do, when_to_stop, how_to_refute)
```

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الجذور والتأسيس - نظرية المعرفة

العلم يتقدم حين يميز بين مراتب اليقين. MICAI يفرض على الوكيل أن يصنف ادعاءه: ثابت، راجح، خلافي، فرضية، أو تركيب. هذا التصنيف جزء من الجواب لا ملحق أخلاقي.

النتيجة التي نستخرجها: كل محاولة لبناء AI معرفي يجب أن تنتقل من وصف المكونات إلى تعريف علاقات السلطة بينها. من يقرر ما يُسترجع؟ من يصدق الدليل؟ من يوقف الأداة؟ من يغير الخطة؟ هذه أسئلة حاكمية.

تمييز حاسم
Cognitive AI ليس LLM + أدوات فقط. هو نظام يملك ذاكرة ذات صلاحية، نموذجا للعالم، محكمة اعتراض، وميزانا لتكلفة التفكير والفعل.

الخطأ الشائع هنا هو اعتبار الزيادة في الحجم بديلا عن البناء المعماري. الحجم يعطي مادة احتمالية؛ الحاكمية تجعلها معرفة قابلة للمساءلة.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الجذور والتأسيس - الوعي كعدسة لا كادعاء

الكتاب يستعمل نظرية التراكب الوعائي للوعي كعدسة توليدية: كيف تتراكب الأوعية؟ كيف يصعد معنى؟ كيف تحكم الغاية؟ لكنه لا يدعي أن الآلة واعية ظاهراتيا.

النتيجة التي نستخرجها: كل محاولة لبناء AI معرفي يجب أن تنتقل من وصف المكونات إلى تعريف علاقات السلطة بينها. من يقرر ما يُسترجع؟ من يصدق الدليل؟ من يوقف الأداة؟ من يغير الخطة؟ هذه أسئلة حاكمية.

تمييز حاسم
Cognitive AI ليس LLM + أدوات فقط. هو نظام يملك ذاكرة ذات صلاحية، نموذجا للعالم، محكمة اعتراض، وميزانا لتكلفة التفكير والفعل.

الخطأ الشائع هنا هو اعتبار الزيادة في الحجم بديلا عن البناء المعماري. الحجم يعطي مادة احتمالية؛ الحاكمية تجعلها معرفة قابلة للمساءلة.

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

الجذور والتأسيس - من المعمارية المعرفية إلى الوكيل

المعماريات المعرفية التاريخية حاولت أن تحدد ما يبقى ثابتا عبر المهام: الذاكرة، آليات القرار، التمثيل، التعلم. وكلاء اللغة أعادوا السؤال لكن بوسيط جديد: نموذج لغوي واسع يصبح قلبا احتماليا داخل بنية أفعال وذاكرة.

النتيجة التي نستخرجها: كل محاولة لبناء AI معرفي يجب أن تنتقل من وصف المكونات إلى تعريف علاقات السلطة بينها. من يقرر ما يُسترجع؟ من يصدق الدليل؟ من يوقف الأداة؟ من يغير الخطة؟ هذه أسئلة حاكمية.

تمييز حاسم
Cognitive AI ليس LLM + أدوات فقط. هو نظام يملك ذاكرة ذات صلاحية، نموذجا للعالم، محكمة اعتراض، وميزانا لتكلفة التفكير والفعل.

الخطأ الشائع هنا هو اعتبار الزيادة في الحجم بديلا عن البناء المعماري. الحجم يعطي مادة احتمالية؛ الحاكمية تجعلها معرفة قابلة للمساءلة.

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

الجذور والتأسيس - ACT-R/Soar/LIDA كجذور

ACT-R يعلّمنا أن البنية والمعرفة منفصلتان، Soar يعلّمنا أن القرار الدوري والتعلم من التعثر قاعدة عامة، LIDA يعلّمنا أن الانتباه ودورة الفعل ليست إضافات بل مركز تنظيم السلوك.

النتيجة التي نستخرجها: كل محاولة لبناء AI معرفي يجب أن تنتقل من وصف المكونات إلى تعريف علاقات السلطة بينها. من يقرر ما يُسترجع؟ من يصدق الدليل؟ من يوقف الأداة؟ من يغير الخطة؟ هذه أسئلة حاكمية.

تمييز حاسم
Cognitive AI ليس LLM + أدوات فقط. هو نظام يملك ذاكرة ذات صلاحية، نموذجا للعالم، محكمة اعتراض، وميزانا لتكلفة التفكير والفعل.

```
Agent_t = <State, Memory, Policy, Tools, World, Goals, Governance>
Governance := Select(what_to_trust, what_to_do, when_to_stop, how_to_refute)
```

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الجذور والتأسيس - CoALA كجسر

CoALA مهم لأنه لا يبدأ من فراغ: يترجم تاريخ الإنتاجات والذاكرة والأفعال إلى لغة وكلاء LLM. لكن الكتاب يرى أن CoALA يحتاج طبقة حاكمية إحالية تقيس الاختطاف وشروط الفشل.

النتيجة التي نستخرجها: كل محاولة لبناء AI معرفي يجب أن تنتقل من وصف المكونات إلى تعريف علاقات السلطة بينها. من يقرر ما يُسترجع؟ من يصدق الدليل؟ من يوقف الأداة؟ من يغير الخطة؟ هذه أسئلة حاكمية.

تمييز حاسم
Cognitive AI ليس LLM + أدوات فقط. هو نظام يملك ذاكرة ذات صلاحية، نموذجا للعالم، محكمة اعتراض، وميزانا لتكلفة التفكير والفعل.

الخطأ الشائع هنا هو اعتبار الزيادة في الحجم بديلا عن البناء المعماري. الحجم يعطي مادة احتمالية؛ الحاكمية تجعلها معرفة قابلة للمساءلة.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

الجذور والتأسيس - التناقض البنيوي

النموذج اللغوي واسع جدا لكنه ليس حاكما بذاته. الأدوات قوية لكنها لا تعرف الغاية. الذاكرة مفيدة لكنها تنقل الخطأ. الاختبار يعطي رقما لكنه قد يخفي سوء الفهم. من هنا يولد MICAI.

النتيجة التي نستخرجها: كل محاولة لبناء AI معرفي يجب أن تنتقل من وصف المكونات إلى تعريف علاقات السلطة بينها. من يقرر ما يُسترجع؟ من يصدق الدليل؟ من يوقف الأداة؟ من يغير الخطة؟ هذه أسئلة حاكمية.

تمييز حاسم
Cognitive AI ليس LLM + أدوات فقط. هو نظام يملك ذاكرة ذات صلاحية، نموذجا للعالم، محكمة اعتراض، وميزانا لتكلفة التفكير والفعل.

الخطأ الشائع هنا هو اعتبار الزيادة في الحجم بديلا عن البناء المعماري. الحجم يعطي مادة احتمالية؛ الحاكمية تجعلها معرفة قابلة للمساءلة.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

الحالة الراهنة - الذاكرة الوكيلية

الدراسات الحديثة تؤكد أن الذاكرة ليست سياقا طويلا فقط. هي حلقة write/manage/read. الذاكرة التي لا تُنقّى تتحول إلى قناة انتقال خطأ، والذاكرة التي تُحذف بعشوائية تفقد الاستمرارية.

التركيب المساعفي: كل مستجد حديث يظل ناقصا إذا لم يوضع داخل MRG. الذاكرة تحتاج وعاء، TTS يحتاج معيار توقف، world model يحتاج إحالة، الأمن يحتاج قياس اختطاف، والتقييم يحتاج شروط فشل لا pass/fail فقط.

ما لا تقوله الأدبيات كفاية
الأبحاث تتقن وصف module منفرد، لكنها أقل حسما في سؤال السلطة المعرفية بين modules. من هنا تظهر إضافة MICAI: هندسة الحاكمية بين المكونات.

```
HijackRisk = dominance(input_i -> decision) * weak_provenance * tool_power
StopThinking when Δconfidence/Δcompute < threshold AND objections_resolved >= k
```

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

الحالة الراهنة - Test-Time Scaling

توسيع الحوسبة وقت الاستدلال يجعل النموذج يفكر أكثر: أخذ عينات، شجرة بحث، self-consistency، verifier، نقد ذاتي. لكن التفكير الأكثر لا يضمن الحاكمية؛ قد يزيد ثقة الخداع إن لم توجد محكمة.

التركيب المساعفي: كل مستجد حديث يظل ناقصا إذا لم يوضع داخل MRG. الذاكرة تحتاج وعاء، TTS يحتاج معيار توقف، world model يحتاج إحالة، الأمن يحتاج قياس اختطاف، والتقييم يحتاج شروط فشل لا pass/fail فقط.

ما لا تقوله الأدبيات كفاية
الأبحاث تتقن وصف module منفرد، لكنها أقل حسما في سؤال السلطة المعرفية بين modules. من هنا تظهر إضافة MICAI: هندسة الحاكمية بين المكونات.

هذا الفصل يضع المستجدات في طبقة تشغيلية: كل بحث حديث يتحول إلى سؤال هندسي داخل HAKIM: ماذا نأخذ؟ كيف نقيسه؟ كيف يفشل؟ كيف نمنعه من اختطاف النظام؟

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

الحالة الراهنة - نماذج العالم

نموذج العالم يمنح الوكيل قدرة محاكاة المستقبل والمضاد للواقع. في MICAI لا يكفي التنبؤ بالحالة؛ يجب ربط الحالة بمصدرها وسببها وقيودها وخطر الفعل.

التركيب المساعفي: كل مستجد حديث يظل ناقصا إذا لم يوضع داخل MRG. الذاكرة تحتاج وعاء، TTS يحتاج معيار توقف، world model يحتاج إحالة، الأمن يحتاج قياس اختطاف، والتقييم يحتاج شروط فشل لا pass/fail فقط.

ما لا تقوله الأدبيات كفاية
الأبحاث تتقن وصف module منفرد، لكنها أقل حسما في سؤال السلطة المعرفية بين modules. من هنا تظهر إضافة MICAI: هندسة الحاكمية بين المكونات.

هذا الفصل يضع المستجدات في طبقة تشغيلية: كل بحث حديث يتحول إلى سؤال هندسي داخل HAKIM: ماذا نأخذ؟ كيف نقيسه؟ كيف يفشل؟ كيف نمنعه من اختطاف النظام؟

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الحالة الراهنة - أمن الوكلاء

prompt injection يكشف مشكلة فلسفية قبل أن يكون ثغرة هندسية: النموذج لا يميز بنيويا بين الأمر والبيان. الحاكمية إذن يجب أن تكون خارجية/فوقية، لا مجرد prompt جديد.

التركيب المساعفي: كل مستجد حديث يظل ناقصا إذا لم يوضع داخل MRG. الذاكرة تحتاج وعاء، TTS يحتاج معيار توقف، world model يحتاج إحالة، الأمن يحتاج قياس اختطاف، والتقييم يحتاج شروط فشل لا pass/fail فقط.

ما لا تقوله الأدبيات كفاية
الأبحاث تتقن وصف module منفرد، لكنها أقل حسما في سؤال السلطة المعرفية بين modules. من هنا تظهر إضافة MICAI: هندسة الحاكمية بين المكونات.

```
HijackRisk = dominance(input_i -> decision) * weak_provenance * tool_power
StopThinking when Δconfidence/Δcompute < threshold AND objections_resolved >= k
```

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

الحالة الراهنة - Coding Agents

SWE-bench نقل التقييم من أسئلة مصطنعة إلى قضايا GitHub. AlphaEvolve نقل الاكتشاف من كتابة حل إلى تطوير حلول عبر evaluators. MICAI يأخذ الدرس: لا معرفة بلا مقيم.

التركيب المساعفي: كل مستجد حديث يظل ناقصا إذا لم يوضع داخل MRG. الذاكرة تحتاج وعاء، TTS يحتاج معيار توقف، world model يحتاج إحالة، الأمن يحتاج قياس اختطاف، والتقييم يحتاج شروط فشل لا pass/fail فقط.

ما لا تقوله الأدبيات كفاية
الأبحاث تتقن وصف module منفرد، لكنها أقل حسما في سؤال السلطة المعرفية بين modules. من هنا تظهر إضافة MICAI: هندسة الحاكمية بين المكونات.

هذا الفصل يضع المستجدات في طبقة تشغيلية: كل بحث حديث يتحول إلى سؤال هندسي داخل HAKIM: ماذا نأخذ؟ كيف نقيسه؟ كيف يفشل؟ كيف نمنعه من اختطاف النظام؟

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

الحالة الراهنة - Small/Hybrid Models

المنافسة لا تحتاج دائما أكبر نموذج. قد تكون البنية الصحيحة: نموذج كبير للتوليد، نماذج صغيرة للرقابة، verifier متخصص، ذاكرة رسومية، ومحرك قواعد حاكم.

التركيب المساعفي: كل مستجد حديث يظل ناقصا إذا لم يوضع داخل MRG. الذاكرة تحتاج وعاء، TTS يحتاج معيار توقف، world model يحتاج إحالة، الأمن يحتاج قياس اختطاف، والتقييم يحتاج شروط فشل لا pass/fail فقط.

ما لا تقوله الأدبيات كفاية
الأبحاث تتقن وصف module منفرد، لكنها أقل حسما في سؤال السلطة المعرفية بين modules. من هنا تظهر إضافة MICAI: هندسة الحاكمية بين المكونات.

هذا الفصل يضع المستجدات في طبقة تشغيلية: كل بحث حديث يتحول إلى سؤال هندسي داخل HAKIM: ماذا نأخذ؟ كيف نقيسه؟ كيف يفشل؟ كيف نمنعه من اختطاف النظام؟

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الحالة الراهنة - الذاكرة الوكيلية

الدراسات الحديثة تؤكد أن الذاكرة ليست سياقا طويلا فقط. هي حلقة write/manage/read. الذاكرة التي لا تُنقّى تتحول إلى قناة انتقال خطأ، والذاكرة التي تُحذف بعشوائية تفقد الاستمرارية.

التركيب المساعفي: كل مستجد حديث يظل ناقصا إذا لم يوضع داخل MRG. الذاكرة تحتاج وعاء، TTS يحتاج معيار توقف، world model يحتاج إحالة، الأمن يحتاج قياس اختطاف، والتقييم يحتاج شروط فشل لا pass/fail فقط.

ما لا تقوله الأدبيات كفاية
الأبحاث تتقن وصف module منفرد، لكنها أقل حسما في سؤال السلطة المعرفية بين modules. من هنا تظهر إضافة MICAI: هندسة الحاكمية بين المكونات.

```
HijackRisk = dominance(input_i -> decision) * weak_provenance * tool_power
StopThinking when Δconfidence/Δcompute < threshold AND objections_resolved >= k
```

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

الحالة الراهنة - Test-Time Scaling

توسيع الحوسبة وقت الاستدلال يجعل النموذج يفكر أكثر: أخذ عينات، شجرة بحث، self-consistency ،verifier، نقد ذاتي. لكن التفكير الأكثر لا يضمن الحاكمية؛ قد يزيد ثقة الخداع إن لم توجد محكمة.

التركيب المساعفي: كل مستجد حديث يظل ناقصا إذا لم يوضع داخل MRG. الذاكرة تحتاج وعاء، TTS يحتاج معيار توقف، world model يحتاج إحالة، الأمن يحتاج قياس اختطاف، والتقييم يحتاج شروط فشل لا pass/fail فقط.

ما لا تقوله الأدبيات كفاية
الأبحاث تتقن وصف module منفرد، لكنها أقل حسما في سؤال السلطة المعرفية بين modules. من هنا تظهر إضافة MICAI: هندسة الحاكمية بين المكونات.

هذا الفصل يضع المستجدات في طبقة تشغيلية: كل بحث حديث يتحول إلى سؤال هندسي داخل HAKIM: ماذا نأخذ؟ كيف نقيسه؟ كيف يفشل؟ كيف نمنعه من اختطاف النظام؟

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

الحالة الراهنة - نماذج العالم

نموذج العالم يمنح الوكيل قدرة محاكاة المستقبل والمضاد للواقع. في MICAI لا يكفي التنبؤ بالحالة؛ يجب ربط الحالة بمصدرها وسببها وقيودها وخطر الفعل.

التركيب المساعفي: كل مستجد حديث يظل ناقصا إذا لم يوضع داخل MRG. الذاكرة تحتاج وعاء، TTS يحتاج معيار توقف، world model يحتاج إحالة، الأمن يحتاج قياس اختطاف، والتقييم يحتاج شروط فشل لا pass/fail فقط.

ما لا تقوله الأدبيات كفاية
الأبحاث تتقن وصف module منفرد، لكنها أقل حسما في سؤال السلطة المعرفية بين modules. من هنا تظهر إضافة MICAI: هندسة الحاكمية بين المكونات.

هذا الفصل يضع المستجدات في طبقة تشغيلية: كل بحث حديث يتحول إلى سؤال هندسي داخل HAKIM: ماذا نأخذ؟ كيف نقيسه؟ كيف يفشل؟ كيف نمنعه من اختطاف النظام؟

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الحالة الراهنة - أمن الوكلاء

prompt injection يكشف مشكلة فلسفية قبل أن يكون ثغرة هندسية: النموذج لا يميز بنيويا بين الأمر والبيان. الحاكمية إذن يجب أن تكون خارجية/فوقية، لا مجرد prompt جديد.

التركيب المساعفي: كل مستجد حديث يظل ناقصا إذا لم يوضع داخل MRG. الذاكرة تحتاج وعاء، TTS يحتاج معيار توقف، world model يحتاج إحالة، الأمن يحتاج قياس اختطاف، والتقييم يحتاج شروط فشل لا pass/fail فقط.

ما لا تقوله الأدبيات كفاية
الأبحاث تتقن وصف module منفرد، لكنها أقل حسما في سؤال السلطة المعرفية بين modules. من هنا تظهر إضافة MICAI: هندسة الحاكمية بين المكونات.

```
HijackRisk = dominance(input_i -> decision) * weak_provenance * tool_power
StopThinking when Δconfidence/Δcompute < threshold AND objections_resolved >= k
```

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

الحالة الراهنة - Coding Agents

SWE-bench نقل التقييم من أسئلة مصطنعة إلى قضايا GitHub. AlphaEvolve نقل الاكتشاف من كتابة حل إلى تطوير حلول عبر evaluators. MICAI يأخذ الدرس: لا معرفة بلا مقيم.

التركيب المساعفي: كل مستجد حديث يظل ناقصا إذا لم يوضع داخل MRG. الذاكرة تحتاج وعاء، TTS يحتاج معيار توقف، world model يحتاج إحالة، الأمن يحتاج قياس اختطاف، والتقييم يحتاج شروط فشل لا pass/fail فقط.

ما لا تقوله الأدبيات كفاية
الأبحاث تتقن وصف module منفرد، لكنها أقل حسما في سؤال السلطة المعرفية بين modules. من هنا تظهر إضافة MICAI: هندسة الحاكمية بين المكونات.

هذا الفصل يضع المستجدات في طبقة تشغيلية: كل بحث حديث يتحول إلى سؤال هندسي داخل HAKIM: ماذا نأخذ؟ كيف نقيسه؟ كيف يفشل؟ كيف نمنعه من اختطاف النظام؟

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

الحالة الراهنة - Small/Hybrid Models

المنافسة لا تحتاج دائما أكبر نموذج. قد تكون البنية الصحيحة: نموذج كبير للتوليد، نماذج صغيرة للرقابة، verifier متخصص، ذاكرة رسومية، ومحرك قواعد حاكم.

التركيب المساعفي: كل مستجد حديث يظل ناقصا إذا لم يوضع داخل MRG. الذاكرة تحتاج وعاء، TTS يحتاج معيار توقف، world model يحتاج إحالة، الأمن يحتاج قياس اختطاف، والتقييم يحتاج شروط فشل لا pass/fail فقط.

ما لا تقوله الأدبيات كفاية
الأبحاث تتقن وصف module منفرد، لكنها أقل حسما في سؤال السلطة المعرفية بين modules. من هنا تظهر إضافة MICAI: هندسة الحاكمية بين المكونات.

هذا الفصل يضع المستجدات في طبقة تشغيلية: كل بحث حديث يتحول إلى سؤال هندسي داخل HAKIM: ماذا نأخذ؟ كيف نقيسه؟ كيف يفشل؟ كيف نمنعه من اختطاف النظام؟

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الحالة الراهنة - الذاكرة الوكيلية

الدراسات الحديثة تؤكد أن الذاكرة ليست سياقا طويلا فقط. هي حلقة write/manage/read. الذاكرة التي لا تُنقّى تتحول إلى قناة انتقال خطأ، والذاكرة التي تُحذف بعشوائية تفقد الاستمرارية.

التركيب المساعفي: كل مستجد حديث يظل ناقصا إذا لم يوضع داخل MRG. الذاكرة تحتاج وعاء، TTS يحتاج معيار توقف، world model يحتاج إحالة، الأمن يحتاج قياس اختطاف، والتقييم يحتاج شروط فشل لا pass/fail فقط.

ما لا تقوله الأدبيات كفاية
الأبحاث تتقن وصف module منفرد، لكنها أقل حسما في سؤال السلطة المعرفية بين modules. من هنا تظهر إضافة MICAI: هندسة الحاكمية بين المكونات.

```
HijackRisk = dominance(input_i -> decision) * weak_provenance * tool_power
StopThinking when Δconfidence/Δcompute < threshold AND objections_resolved >= k
```

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

الحالة الراهنة - Test-Time Scaling

توسيع الحوسبة وقت الاستدلال يجعل النموذج يفكر أكثر: أخذ عينات، شجرة بحث، self-consistency، verifier، نقد ذاتي. لكن التفكير الأكثر لا يضمن الحاكمية؛ قد يزيد ثقة الخداع إن لم توجد محكمة.

التركيب المساعفي: كل مستجد حديث يظل ناقصا إذا لم يوضع داخل MRG. الذاكرة تحتاج وعاء، TTS يحتاج معيار توقف، world model يحتاج إحالة، الأمن يحتاج قياس اختطاف، والتقييم يحتاج شروط فشل لا pass/fail فقط.

ما لا تقوله الأدبيات كفاية
الأبحاث تتقن وصف module منفرد، لكنها أقل حسما في سؤال السلطة المعرفية بين modules. من هنا تظهر إضافة MICAI: هندسة الحاكمية بين المكونات.

هذا الفصل يضع المستجدات في طبقة تشغيلية: كل بحث حديث يتحول إلى سؤال هندسي داخل HAKIM: ماذا نأخذ؟ كيف نقيسه؟ كيف يفشل؟ كيف نمنعه من اختطاف النظام؟

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

الحالة الراهنة - نماذج العالم

نموذج العالم يمنح الوكيل قدرة محاكاة المستقبل والمضاد للواقع. في MICAI لا يكفي التنبؤ بالحالة؛ يجب ربط الحالة بمصدرها وسببها وقيودها وخطر الفعل.

التركيب المساعفي: كل مستجد حديث يظل ناقصا إذا لم يوضع داخل MRG. الذاكرة تحتاج وعاء، TTS يحتاج معيار توقف، world model يحتاج إحالة، الأمن يحتاج قياس اختطاف، والتقييم يحتاج شروط فشل لا pass/fail فقط.

ما لا تقوله الأدبيات كفاية
الأبحاث تتقن وصف module منفرد، لكنها أقل حسما في سؤال السلطة المعرفية بين modules. من هنا تظهر إضافة MICAI: هندسة الحاكمية بين المكونات.

هذا الفصل يضع المستجدات في طبقة تشغيلية: كل بحث حديث يتحول إلى سؤال هندسي داخل HAKIM: ماذا نأخذ؟ كيف نقيسه؟ كيف يفشل؟ كيف نمنعه من اختطاف النظام؟

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الحالة الراهنة - أمن الوكلاء

prompt injection يكشف مشكلة فلسفية قبل أن يكون ثغرة هندسية: النموذج لا يميز بنيويا بين الأمر والبيان. الحاكمية إذن يجب أن تكون خارجية/فوقية، لا مجرد prompt جديد.

التركيب المساعفي: كل مستجد حديث يظل ناقصا إذا لم يوضع داخل MRG. الذاكرة تحتاج وعاء، TTS يحتاج معيار توقف، world model يحتاج إحالة، الأمن يحتاج قياس اختطاف، والتقييم يحتاج شروط فشل لا pass/fail فقط.

ما لا تقوله الأدبيات كفاية
الأبحاث تتقن وصف module منفرد، لكنها أقل حسما في سؤال السلطة المعرفية بين modules. من هنا تظهر إضافة MICAI: هندسة الحاكمية بين المكونات.

```
HijackRisk = dominance(input_i -> decision) * weak_provenance * tool_power
StopThinking when Δconfidence/Δcompute < threshold AND objections_resolved >= k
```

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

الحالة الراهنة - Coding Agents

SWE-bench نقل التقييم من أسئلة مصطنعة إلى قضايا GitHub. AlphaEvolve نقل الاكتشاف من كتابة حل إلى تطوير حلول عبر evaluators. MICAI يأخذ الدرس: لا معرفة بلا مقيم.

التركيب المساعفي: كل مستجد حديث يظل ناقصا إذا لم يوضع داخل MRG. الذاكرة تحتاج وعاء، TTS يحتاج معيار توقف، world model يحتاج إحالة، الأمن يحتاج قياس اختطاف، والتقييم يحتاج شروط فشل لا pass/fail فقط.

ما لا تقوله الأدبيات كفاية
الأبحاث تتقن وصف module منفرد، لكنها أقل حسما في سؤال السلطة المعرفية بين modules. من هنا تظهر إضافة MICAI: هندسة الحاكمية بين المكونات.

هذا الفصل يضع المستجدات في طبقة تشغيلية: كل بحث حديث يتحول إلى سؤال هندسي داخل HAKIM: ماذا نأخذ؟ كيف نقيسه؟ كيف يفشل؟ كيف نمنعه من اختطاف النظام؟

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

الحالة الراهنة - Small/Hybrid Models

المنافسة لا تحتاج دائما أكبر نموذج. قد تكون البنية الصحيحة: نموذج كبير للتوليد، نماذج صغيرة للرقابة، verifier متخصص، ذاكرة رسومية، ومحرك قواعد حاكم.

التركيب المساعفي: كل مستجد حديث يظل ناقصا إذا لم يوضع داخل MRG. الذاكرة تحتاج وعاء، TTS يحتاج معيار توقف، world model يحتاج إحالة، الأمن يحتاج قياس اختطاف، والتقييم يحتاج شروط فشل لا pass/fail فقط.

ما لا تقوله الأدبيات كفاية
الأبحاث تتقن وصف module منفرد، لكنها أقل حسما في سؤال السلطة المعرفية بين modules. من هنا تظهر إضافة MICAI: هندسة الحاكمية بين المكونات.

هذا الفصل يضع المستجدات في طبقة تشغيلية: كل بحث حديث يتحول إلى سؤال هندسي داخل HAKIM: ماذا نأخذ؟ كيف نقيسه؟ كيف يفشل؟ كيف نمنعه من اختطاف النظام؟

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

MICAI - الحاكمية الإحالية MRG

الحاكمية الإحالية MRG

قدرة النظام على ترتيب العلاقة بين الادعاء والدليل والسياق والمصدر والفعل والغاية وشروط الفشل.

داخل MICAI، لا تُعامل المفاهيم كاستعارات. الوعاء يصبح متغيرا في الذاكرة؛ الحاكمية تصبح دالة اختيار؛ الاختطاف يصبح مؤشرا قابلا للقياس؛ MCD يصبح دالة تقييم لعمق الفرق لا لكثرة الإجابة.

الحلقة العامة: يواجه الوكيل سؤالا، يحدد وعاءه، يسترجع ذاكرة ذات صلاحية، يبني فرضيات، يطلب الدليل، يولد اعتراضات، يقترح فعلا أو تجربة، ثم يسجل كل ذلك في جواز معرفي.

قاعدة مساعف
لا يكون الفرق معرفيا حتى يدخل وعاء صالحا؛ ولا يكون الوعاء ذكيا حتى يُحكم؛ ولا تكون الحاكمية علمية حتى تقبل الهدم.

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

MICAI - الوعاء المعرفي Ω

الوعاء المعرفي Ω

حاوية تشغيلية: مجال، سياق، ذاكرة، هدف، زمن، أداة، أو نمط تمثيل يؤثر في معنى الفرق.

داخل MICAI، لا تُعامل المفاهيم كاستعارات. الوعاء يصبح متغيرا في الذاكرة؛ الحاكمية تصبح دالة اختيار؛ الاختطاف يصبح مؤشرا قابلا للقياس؛ MCD يصبح دالة تقييم لعمق الفرق لا لكثرة الإجابة.

الحلقة العامة: يواجه الوكيل سؤالا، يحدد وعاءه، يسترجع ذاكرة ذات صلاحية، يبني فرضيات، يطلب الدليل، يولد اعتراضات، يقترح فعلا أو تجربة، ثم يسجل كل ذلك في جواز معرفي.

```
Δ := difference that changes expectation/action
Ω := vessel/domain/context/time/goal/tool
Meaning(Δ) = Bind(Δ, Ω, Memory, Goal)
Govern(Meaning) = rank by evidence, risk, actionability, falsifiability
```

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

MICAI - الفرق الفاعل Δ

الفرق الفاعل Δ

وحدة أولية للمعنى العملي: اختلاف يغيّر التوقع أو القرار أو القدرة على الفعل.

داخل MICAI، لا تُعامل المفاهيم كاستعارات. الوعاء يصبح متغيرا في الذاكرة؛ الحاكمية تصبح دالة اختيار؛ الاختطاف يصبح مؤشرا قابلا للقياس؛ MCD يصبح دالة تقييم لعمق الفرق لا لكثرة الإجابة.

الحلقة العامة: يواجه الوكيل سؤالا، يحدد وعاءه، يسترجع ذاكرة ذات صلاحية، يبني فرضيات، يطلب الدليل، يولد اعتراضات، يقترح فعلا أو تجربة، ثم يسجل كل ذلك في جواز معرفي.

قاعدة مساعف
لا يكون الفرق معرفيا حتى يدخل وعاء صالحا، ولا يكون الوعاء ذكيا حتى يُحكم، ولا تكون الحاكمية علمية حتى تقبل الهدم.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

MICAI - MCD

MCD

Meta-Complexity Differentiation: مقياس قدرة النظام على توليد فروق عميقة، ربطها، نقدها، ونقلها بين الأوعية.

داخل MICAI، لا تُعامل المفاهيم كاستعارات. الوعاء يصبح متغيرا في الذاكرة؛ الحاكمية تصبح دالة اختيار؛ الاختطاف يصبح مؤشرا قابلا للقياس؛ MCD يصبح دالة تقييم لعمق الفرق لا لكثرة الإجابة.

الحلقة العامة: يواجه الوكيل سؤالا، يحدد وعاءه، يسترجع ذاكرة ذات صلاحية، يبني فرضيات، يطلب الدليل، يولد اعتراضات، يقترح فعلا أو تجربة، ثم يسجل كل ذلك في جواز معرفي.

```
Δ := difference that changes expectation/action
Ω := vessel/domain/context/time/goal/tool
Meaning(Δ) = Bind(Δ, Ω, Memory, Goal)
Govern(Meaning) = rank by evidence, risk, actionability, falsifiability
```

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

MICAI - HAKIM

HAKIM

بروتوكول حاكمية: Hypothesis, Audit, Knowledge, Intervention, Metrics.

داخل MICAI، لا تُعامل المفاهيم كاستعارات. الوعاء يصبح متغيرا في الذاكرة؛ الحاكمية تصبح دالة اختيار؛ الاختطاف يصبح مؤشرا قابلا للقياس؛ MCD يصبح دالة تقييم لعمق الفرق لا لكثرة الإجابة.

الحلقة العامة: يواجه الوكيل سؤالا، يحدد وعاءه، يسترجع ذاكرة ذات صلاحية، يبني فرضيات، يطلب الدليل، يولد اعتراضات، يقترح فعلا أو تجربة، ثم يسجل كل ذلك في جواز معرفي.

قاعدة مساعف
لا يكون الفرق معرفيا حتى يدخل وعاء صالحا؛ ولا يكون الوعاء ذكيا حتى يُحكم؛ ولا تكون الحاكمية علمية حتى تقبل الهدم.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

MICAI - الاختطاف الوعائي

الاختطاف الوعائي

صعود وعاء جزئي إلى مقام الحاكمية العامة فيشوّه القرار: ذاكرة خاطئة، مصدر ملوث، هدف ضيق، prompt injection.

داخل MICAI، لا تُعامل المفاهيم كاستعارات. الوعاء يصبح متغيرا في الذاكرة؛ الحاكمية تصبح دالة اختيار؛ الاختطاف يصبح مؤشرا قابلا للقياس؛ MCD يصبح دالة تقييم لعمق الفرق لا لكثرة الإجابة.

الحلقة العامة: يواجه الوكيل سؤالا، يحدد وعاءه، يسترجع ذاكرة ذات صلاحية، يبني فرضيات، يطلب الدليل، يولد اعتراضات، يقترح فعلا أو تجربة، ثم يسجل كل ذلك في جواز معرفي.

```
Δ := difference that changes expectation/action
Ω := vessel/domain/context/time/goal/tool
Meaning(Δ) = Bind(Δ, Ω, Memory, Goal)
Govern(Meaning) = rank by evidence, risk, actionability, falsifiability
```

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

MICAI - الإثبات المحمول معرفيا

الإثبات المحمول معرفيا

كل جواب يحمل ادعاءه ودليله واعتراضه وشروط فشله وسجل القرار الذي أنتجه.

داخل MICAI، لا تُعامل المفاهيم كاستعارات. الوعاء يصبح متغيرا في الذاكرة؛ الحاكمية تصبح دالة اختيار؛ الاختطاف يصبح مؤشرا قابلا للقياس؛ MCD يصبح دالة تقييم لعمق الفرق لا لكثرة الإجابة.

الحلقة العامة: يواجه الوكيل سؤالا، يحدد وعاءه، يسترجع ذاكرة ذات صلاحية، يبني فرضيات، يطلب الدليل، يولد اعتراضات، يقترح فعلا أو تجربة، ثم يسجل كل ذلك في جواز معرفي.

قاعدة مساعف
لا يكون الفرق معرفيا حتى يدخل وعاء صالحا؛ ولا يكون الوعاء ذكيا حتى يُحكم؛ ولا تكون الحاكمية علمية حتى تقبل الهدم.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

MICAI - الذاكرة الوعائية

الذاكرة الوعائية

ذاكرة لا تخزن النص فقط، بل تخزن المصدر، الصلاحية، المجال، الاعتراضات، التجارب، ونمط الفشل.

داخل MICAI، لا تُعامل المفاهيم كاستعارات. الوعاء يصبح متغيرا في الذاكرة؛ الحاكمية تصبح دالة اختيار؛ الاختطاف يصبح مؤشرا قابلا للقياس؛ MCD يصبح دالة تقييم لعمق الفرق لا لكثرة الإجابة.

الحلقة العامة: يواجه الوكيل سؤالا، يحدد وعاءه، يسترجع ذاكرة ذات صلاحية، يبني فرضيات، يطلب الدليل، يولد اعتراضات، يقترح فعلا أو تجربة، ثم يسجل كل ذلك في جواز معرفي.

```
Δ := difference that changes expectation/action
Ω := vessel/domain/context/time/goal/tool
Meaning(Δ) = Bind(Δ, Ω, Memory, Goal)
Govern(Meaning) = rank by evidence, risk, actionability, falsifiability
```

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

MICAI - الحاكمية فوق الحاكمية

الحاكمية فوق الحاكمية

طبقة تراقب قرارات طبقة التحكم نفسها، وتكشف انحراف المعايير مع الزمن.

داخل MICAI، لا تُعامل المفاهيم كاستعارات. الوعاء يصبح متغيرا في الذاكرة؛ الحاكمية تصبح دالة اختيار؛ الاختطاف يصبح مؤشرا قابلا للقياس؛ MCD يصبح دالة تقييم لعمق الفرق لا لكثرة الإجابة.

الحلقة العامة: يواجه الوكيل سؤالا، يحدد وعاءه، يسترجع ذاكرة ذات صلاحية، يبني فرضيات، يطلب الدليل، يولد اعتراضات، يقترح فعلا أو تجربة، ثم يسجل كل ذلك في جواز معرفي.

قاعدة مساعف
لا يكون الفرق معرفيا حتى يدخل وعاء صالحا؛ ولا يكون الوعاء ذكيا حتى يُحكم؛ ولا تكون الحاكمية علمية حتى تقبل الهدم.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

MICAI - محكمة الهدم

محكمة الهدم

آلية منهجية لتوليد أقوى اعتراضات ثم تعديل النموذج لا الدفاع عنه بلاغيا.

داخل MICAI، لا تُعامل المفاهيم كاستعارات. الوعاء يصبح متغيرا في الذاكرة؛ الحاكمية تصبح دالة اختيار؛ الاختطاف يصبح مؤشرا قابلا للقياس؛ MCD يصبح دالة تقييم لعمق الفرق لا لكثرة الإجابة.

الحلقة العامة: يواجه الوكيل سؤالا، يحدد وعاءه، يسترجع ذاكرة ذات صلاحية، يبني فرضيات، يطلب الدليل، يولد اعتراضات، يقترح فعلا أو تجربة، ثم يسجل كل ذلك في جواز معرفي.

```
Δ := difference that changes expectation/action
Ω := vessel/domain/context/time/goal/tool
Meaning(Δ) = Bind(Δ, Ω, Memory, Goal)
Govern(Meaning) = rank by evidence, risk, actionability, falsifiability
```

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

MICAI - الحاكمية الإحالية MRG

الحاكمية الإحالية MRG

قدرة النظام على ترتيب العلاقة بين الادعاء والدليل والسياق والمصدر والفعل والغاية وشروط الفشل.

داخل MICAI، لا تُعامل المفاهيم كاستعارات. الوعاء يصبح متغيرا في الذاكرة؛ الحاكمية تصبح دالة اختيار؛ الاختطاف يصبح مؤشرا قابلا للقياس؛ MCD يصبح دالة تقييم لعمق الفرق لا لكثرة الإجابة.

الحلقة العامة: يواجه الوكيل سؤالا، يحدد وعاءه، يسترجع ذاكرة ذات صلاحية، يبني فرضيات، يطلب الدليل، يولد اعتراضات، يقترح فعلا أو تجربة، ثم يسجل كل ذلك في جواز معرفي.

قاعدة مساعف
لا يكون الفرق معرفيا حتى يدخل وعاء صالحا؛ ولا يكون الوعاء ذكيا حتى يُحكم؛ ولا تكون الحاكمية علمية حتى تقبل الهدم.

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

MICAI - الوعاء المعرفي Ω

الوعاء المعرفي Ω

حاوية تشغيلية: مجال، سياق، ذاكرة، هدف، زمن، أداة، أو نمط تمثيل يؤثر في معنى الفرق.

داخل MICAI، لا تُعامل المفاهيم كاستعارات. الوعاء يصبح متغيرا في الذاكرة؛ الحاكمية تصبح دالة اختيار؛ الاختطاف يصبح مؤشرا قابلا للقياس؛ MCD يصبح دالة تقييم لعمق الفرق لا لكثرة الإجابة.

الحلقة العامة: يواجه الوكيل سؤالا، يحدد وعاءه، يسترجع ذاكرة ذات صلاحية، يبني فرضيات، يطلب الدليل، يولد اعتراضات، يقترح فعلا أو تجربة، ثم يسجل كل ذلك في جواز معرفي.

```
Δ := difference that changes expectation/action
Ω := vessel/domain/context/time/goal/tool
Meaning(Δ) = Bind(Δ, Ω, Memory, Goal)
Govern(Meaning) = rank by evidence, risk, actionability, falsifiability
```

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

MICAI - الفرق الفاعل Δ

الفرق الفاعل Δ

وحدة أولية للمعنى العملي: اختلاف يغيّر التوقع أو القرار أو القدرة على الفعل.

داخل MICAI، لا تُعامل المفاهيم كاستعارات. الوعاء يصبح متغيرا في الذاكرة؛ الحاكمية تصبح دالة اختيار؛ الاختطاف يصبح مؤشرا قابلا للقياس؛ MCD يصبح دالة تقييم لعمق الفرق لا لكثرة الإجابة.

الحلقة العامة: يواجه الوكيل سؤالا، يحدد وعاءه، يسترجع ذاكرة ذات صلاحية، يبني فرضيات، يطلب الدليل، يولد اعتراضات، يقترح فعلا أو تجربة، ثم يسجل كل ذلك في جواز معرفي.

قاعدة مساعف
لا يكون الفرق معرفيا حتى يدخل وعاء صالحا، ولا يكون الوعاء ذكيا حتى يُحكم، ولا تكون الحاكمية علمية حتى تقبل الهدم.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

MICAI - MCD

MCD

Meta-Complexity Differentiation: مقياس قدرة النظام على توليد فروق عميقة، ربطها، نقدها، ونقلها بين الأوعية.

داخل MICAI، لا تُعامل المفاهيم كاستعارات. الوعاء يصبح متغيرا في الذاكرة؛ الحاكمية تصبح دالة اختيار؛ الاختطاف يصبح مؤشرا قابلا للقياس؛ MCD يصبح دالة تقييم لعمق الفرق لا لكثرة الإجابة.

الحلقة العامة: يواجه الوكيل سؤالا، يحدد وعاءه، يسترجع ذاكرة ذات صلاحية، يبني فرضيات، يطلب الدليل، يولد اعتراضات، يقترح فعلا أو تجربة، ثم يسجل كل ذلك في جواز معرفي.

```
Δ := difference that changes expectation/action
Ω := vessel/domain/context/time/goal/tool
Meaning(Δ) = Bind(Δ, Ω, Memory, Goal)
Govern(Meaning) = rank by evidence, risk, actionability, falsifiability
```

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

MICAI - HAKIM

HAKIM

بروتوكول حاكمية: Hypothesis, Audit, Knowledge, Intervention, Metrics.

داخل MICAI، لا تُعامل المفاهيم كاستعارات. الوعاء يصبح متغيرا في الذاكرة؛ الحاكمية تصبح دالة اختيار؛ الاختطاف يصبح مؤشرا قابلا للقياس؛ MCD يصبح دالة تقييم لعمق الفرق لا لكثرة الإجابة.

الحلقة العامة: يواجه الوكيل سؤالا، يحدد وعاءه، يسترجع ذاكرة ذات صلاحية، يبني فرضيات، يطلب الدليل، يولد اعتراضات، يقترح فعلا أو تجربة، ثم يسجل كل ذلك في جواز معرفي.

قاعدة مساعف
لا يكون الفرق معرفيا حتى يدخل وعاء صالحا؛ ولا يكون الوعاء ذكيا حتى يُحكم؛ ولا تكون الحاكمية علمية حتى تقبل الهدم.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

MICAI - الاختطاف الوعائي

الاختطاف الوعائي

صعود وعاء جزئي إلى مقام الحاكمية العامة فيشوّه القرار: ذاكرة خاطئة، مصدر ملوث، هدف ضيق، prompt injection.

داخل MICAI، لا تُعامل المفاهيم كاستعارات. الوعاء يصبح متغيرا في الذاكرة؛ الحاكمية تصبح دالة اختيار؛ الاختطاف يصبح مؤشرا قابلا للقياس؛ MCD يصبح دالة تقييم لعمق الفرق لا لكثرة الإجابة.

الحلقة العامة: يواجه الوكيل سؤالا، يحدد وعاءه، يسترجع ذاكرة ذات صلاحية، يبني فرضيات، يطلب الدليل، يولد اعتراضات، يقترح فعلا أو تجربة، ثم يسجل كل ذلك في جواز معرفي.

```
Δ := difference that changes expectation/action
Ω := vessel/domain/context/time/goal/tool
Meaning(Δ) = Bind(Δ, Ω, Memory, Goal)
Govern(Meaning) = rank by evidence, risk, actionability, falsifiability
```

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

MICAI - الإثبات المحمول معرفيا

الإثبات المحمول معرفيا

كل جواب يحمل ادعاءه ودليله واعتراضه وشروط فشله وسجل القرار الذي أنتجه.

داخل MICAI، لا تُعامل المفاهيم كاستعارات. الوعاء يصبح متغيرا في الذاكرة؛ الحاكمية تصبح دالة اختيار؛ الاختطاف يصبح مؤشرا قابلا للقياس؛ MCD يصبح دالة تقييم لعمق الفرق لا لكثرة الإجابة.

الحلقة العامة: يواجه الوكيل سؤالا، يحدد وعاءه، يسترجع ذاكرة ذات صلاحية، يبني فرضيات، يطلب الدليل، يولد اعتراضات، يقترح فعلا أو تجربة، ثم يسجل كل ذلك في جواز معرفي.

قاعدة مساعف
لا يكون الفرق معرفيا حتى يدخل وعاء صالحا؛ ولا يكون الوعاء ذكيا حتى يُحكم؛ ولا تكون الحاكمية علمية حتى تقبل الهدم.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

MICAI - الذاكرة الوعائية

الذاكرة الوعائية

ذاكرة لا تخزن النص فقط، بل تخزن المصدر، الصلاحية، المجال، الاعتراضات، التجارب، ونمط الفشل.

داخل MICAI، لا تُعامل المفاهيم كاستعارات. الوعاء يصبح متغيرا في الذاكرة؛ الحاكمية تصبح دالة اختيار؛ الاختطاف يصبح مؤشرا قابلا للقياس؛ MCD يصبح دالة تقييم لعمق الفرق لا لكثرة الإجابة.

الحلقة العامة: يواجه الوكيل سؤالا، يحدد وعاءه، يسترجع ذاكرة ذات صلاحية، يبني فرضيات، يطلب الدليل، يولد اعتراضات، يقترح فعلا أو تجربة، ثم يسجل كل ذلك في جواز معرفي.

```
Δ := difference that changes expectation/action
Ω := vessel/domain/context/time/goal/tool
Meaning(Δ) = Bind(Δ, Ω, Memory, Goal)
Govern(Meaning) = rank by evidence, risk, actionability, falsifiability
```

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

MICAI - الحاكمية فوق الحاكمية

الحاكمية فوق الحاكمية

طبقة تراقب قرارات طبقة التحكم نفسها، وتكشف انحراف المعايير مع الزمن.

داخل MICAI، لا تُعامل المفاهيم كاستعارات. الوعاء يصبح متغيرا في الذاكرة؛ الحاكمية تصبح دالة اختيار؛ الاختطاف يصبح مؤشرا قابلا للقياس؛ MCD يصبح دالة تقييم لعمق الفرق لا لكثرة الإجابة.

الحلقة العامة: يواجه الوكيل سؤالا، يحدد وعاءه، يسترجع ذاكرة ذات صلاحية، يبني فرضيات، يطلب الدليل، يولد اعتراضات، يقترح فعلا أو تجربة، ثم يسجل كل ذلك في جواز معرفي.

قاعدة مساعف
لا يكون الفرق معرفيا حتى يدخل وعاء صالحا؛ ولا يكون الوعاء ذكيا حتى يُحكم؛ ولا تكون الحاكمية علمية حتى تقبل الهدم.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

MICAI - محكمة الهدم

محكمة الهدم

آلية منهجية لتوليد أقوى اعتراضات ثم تعديل النموذج لا الدفاع عنه بلاغيا.

داخل MICAI، لا تُعامل المفاهيم كاستعارات. الوعاء يصبح متغيرا في الذاكرة؛ الحاكمية تصبح دالة اختيار؛ الاختطاف يصبح مؤشرا قابلا للقياس؛ MCD يصبح دالة تقييم لعمق الفرق لا لكثرة الإجابة.

الحلقة العامة: يواجه الوكيل سؤالا، يحدد وعاءه، يسترجع ذاكرة ذات صلاحية، يبني فرضيات، يطلب الدليل، يولد اعتراضات، يقترح فعلا أو تجربة، ثم يسجل كل ذلك في جواز معرفي.

```
Δ := difference that changes expectation/action
Ω := vessel/domain/context/time/goal/tool
Meaning(Δ) = Bind(Δ, Ω, Memory, Goal)
Govern(Meaning) = rank by evidence, risk, actionability, falsifiability
```

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

MICAI - الحاكمية الإحالية MRG

الحاكمية الإحالية MRG

قدرة النظام على ترتيب العلاقة بين الادعاء والدليل والسياق والمصدر والفعل والغاية وشروط الفشل.

داخل MICAI، لا تُعامل المفاهيم كاستعارات. الوعاء يصبح متغيرا في الذاكرة؛ الحاكمية تصبح دالة اختيار؛ الاختطاف يصبح مؤشرا قابلا للقياس؛ MCD يصبح دالة تقييم لعمق الفرق لا لكثرة الإجابة.

الحلقة العامة: يواجه الوكيل سؤالا، يحدد وعاءه، يسترجع ذاكرة ذات صلاحية، يبني فرضيات، يطلب الدليل، يولد اعتراضات، يقترح فعلا أو تجربة، ثم يسجل كل ذلك في جواز معرفي.

قاعدة مساعف
لا يكون الفرق معرفيا حتى يدخل وعاء صالحا؛ ولا يكون الوعاء ذكيا حتى يُحكم؛ ولا تكون الحاكمية علمية حتى تقبل الهدم.

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

MICAI - الوعاء المعرفي Ω

الوعاء المعرفي Ω

حاوية تشغيلية: مجال، سياق، ذاكرة، هدف، زمن، أداة، أو نمط تمثيل يؤثر في معنى الفرق.

داخل MICAI، لا تُعامل المفاهيم كاستعارات. الوعاء يصبح متغيرا في الذاكرة؛ الحاكمية تصبح دالة اختيار؛ الاختطاف يصبح مؤشرا قابلا للقياس؛ MCD يصبح دالة تقييم لعمق الفرق لا لكثرة الإجابة.

الحلقة العامة: يواجه الوكيل سؤالا، يحدد وعاءه، يسترجع ذاكرة ذات صلاحية، يبني فرضيات، يطلب الدليل، يولد اعتراضات، يقترح فعلا أو تجربة، ثم يسجل كل ذلك في جواز معرفي.

```
Δ := difference that changes expectation/action
Ω := vessel/domain/context/time/goal/tool
Meaning(Δ) = Bind(Δ, Ω, Memory, Goal)
Govern(Meaning) = rank by evidence, risk, actionability, falsifiability
```

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

الصياغة الرياضية - دالة الذكاء المعرفي

نعرّف الذكاء المعرفي لا كاحتمال token بل كدالة على الأوعية والذاكرة والأدلة والفرضيات والاعتراضات والأفعال وشروط الفشل.

```
I_c(t)=G_θ(V_t, M_t, E_t, H_t, A_t, R_t, Φ_t)
Goal: maximize ValidActionableKnowledge, minimize Hijack + Contradiction + Risk
```

هذه الصياغات أولية وليست نهائية. قيمتها أنها تحول الفلسفة إلى برنامج: يمكن قياس المتغيرات، بناء simulation، ومقارنة baseline مع HAKIM-controller.

شرط الرصانة
كل معادلة هنا يجب أن تمر بتجربة: ablation، حساسية، توزيع حالات، ومقارنة مع وكيل بلا حاكمية. ما لا يتحسن بعد القياس يُحذف أو يُعدّل.

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

الصياغة الرياضية - MCD

مقياس تمايز التعقيد الفوقي يجمع: تنوع الفرضيات، تعدد الأطر، عمق السببية، قوة المضاد للواقع، نقل البنية، ومقاومة الهشاشة.

```
MCD = D_hyp + D_frame + D_causal + D_counterfactual + D_transfer - F_fragility - N_noise
```

هذه الصياغات أولية وليست نهائية. قيمتها أنها تحول الفلسفة إلى برنامج: يمكن قياس المتغيرات، بناء simulation، ومقارنة baseline مع HAKIM-controller.

شرط الرصانة
كل معادلة هنا يجب أن تمر بتجربة: ablation، حساسية، توزيع حالات، ومقارنة مع وكيل بلا حاكمية. ما لا يتحسن بعد القياس يُحذف أو يُعدّل.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

الصياغة الرياضية - مؤشر الاختطاف

نقيس سيطرة مدخل أو ذاكرة أو أداة على القرار عبر حساسية دالة الحاكمية لذلك المتغير مقارنة ببقية المتغيرات.

```
Hijack(z_i)= |∂G/∂z_i| / (ε + Σ_{j≠i}|∂G/∂z_j|)
Hijack if Hijack(z_i)>τ and provenance(z_i)<ρ
```

هذه الصياغات أولية وليست نهائية. قيمتها أنها تحول الفلسفة إلى برنامج: يمكن قياس المتغيرات، بناء simulation، ومقارنة baseline مع HAKIM-controller.

شرط الرصانة
كل معادلة هنا يجب أن تمر بتجربة: ablation، حساسية، توزيع حالات، ومقارنة مع وكيل بلا حاكمية. ما لا يتحسن بعد القياس يُحذف أو يُعدّل.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

الصياغة الرياضية - ميزانية التفكير

لا نريد تفكيرا طويلا دائما، نريد تفكيرا مناسبا. إذن نحسب العائد الهامشي لكل خطوة reasoning مقابل التكلفة والمخاطر.

```
Utility(step)= Δconfidence + Δfalsifiability + Δcoverage - λ_cost*compute - λ_risk*action_risk
```

هذه الصياغات أولية وليست نهائية. قيمتها أنها تحول الفلسفة إلى برنامج: يمكن قياس المتغيرات، بناء simulation، ومقارنة baseline مع HAKIM-controller.

شرط الرصانة
كل معادلة هنا يجب أن تمر بتجربة: ablation، حساسية، توزيع حالات، ومقارنة مع وكيل بلا حاكمية. ما لا يتحسن بعد القياس يُحذف أو يُعدّل.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

الصياغة الرياضية - ذاكرة وعائية

كل ذكرى عقدة لها محتوى ومصدر وزمن وصلاحية ومجال واعتراض وقيمة وفشل سابق. الاسترجاع وزن متعدد لا تشابه vector فقط.

```
m_i=(content, source, time, scope, confidence, decay, contradiction, utility, failure_tags)
Retrieve(q)=argmax[sem+ref+causal+freshness-risk]
```

هذه الصياغات أولية وليست نهائية. قيمتها أنها تحول الفلسفة إلى برنامج: يمكن قياس المتغيرات، بناء simulation، ومقارنة baseline مع HAKIM-controller.

شرط الرصانة
كل معادلة هنا يجب أن تمر بتجربة: ablation، حساسية، توزيع حالات، ومقارنة مع وكيل بلا حاكمية. ما لا يتحسن بعد القياس يُحذف أو يُعدّل.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

الصياغة الرياضية - إثبات محمول

كل جواب يُعبّر عنه ككائن بيانات: Trace، FailureConditions، Tests، Objections، Reasoning، Evidence، Claim.

```
AnswerPacket = {claim, evidence[], reasoning_trace, objections[], tests[], failure_conditions[],
uncertainty}
```

هذه الصياغات أولية وليست نهائية. قيمتها أنها تحول الفلسفة إلى برنامج: يمكن قياس المتغيرات، بناء simulation، ومقارنة baseline مع HAKIM-controller.

شرط الرصانة
كل معادلة هنا يجب أن تمر بتجربة: ablation، حساسية، توزيع حالات، ومقارنة مع وكيل بلا حاكمية. ما لا يتحسن بعد القياس يُحذف أو يُعدّل.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الصياغة الرياضية - دالة الذكاء المعرفي

نعرّف الذكاء المعرفي لا كاحتمال token بل كدالة على الأوعية والذاكرة والأدلة والفرضيات والاعتراضات والأفعال وشروط الفشل.

```
I_c(t)=G_θ(V_t, M_t, E_t, H_t, A_t, R_t, Φ_t)
Goal: maximize ValidActionableKnowledge, minimize Hijack + Contradiction + Risk
```

هذه الصياغات أولية وليست نهائية. قيمتها أنها تحول الفلسفة إلى برنامج: يمكن قياس المتغيرات، بناء simulation، ومقارنة baseline مع HAKIM-controller.

شرط الرصانة
كل معادلة هنا يجب أن تمر بتجربة: ablation، حساسية، توزيع حالات، ومقارنة مع وكيل بلا حاكمية. ما لا يتحسن بعد القياس يُحذف أو يُعدّل.

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

الصياغة الرياضية - MCD

مقياس تمايز التعقيد الفوقي يجمع: تنوع الفرضيات، تعدد الأطر، عمق السببية، قوة المضاد للواقع، نقل البنية، ومقاومة الهشاشة.

```
MCD = D_hyp + D_frame + D_causal + D_counterfactual + D_transfer - F_fragility - N_noise
```

هذه الصياغات أولية وليست نهائية. قيمتها أنها تحول الفلسفة إلى برنامج: يمكن قياس المتغيرات، بناء simulation، ومقارنة baseline مع HAKIM-controller.

شرط الرصانة
كل معادلة هنا يجب أن تمر بتجربة: ablation، حساسية، توزيع حالات، ومقارنة مع وكيل بلا حاكمية. ما لا يتحسن بعد القياس يُحذف أو يُعدّل.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

الصياغة الرياضية - مؤشر الاختطاف

نقيس سيطرة مدخل أو ذاكرة أو أداة على القرار عبر حساسية دالة الحاكمية لذلك المتغير مقارنة ببقية المتغيرات.

```
Hijack(z_i)= |∂G/∂z_i| / (ε + Σ_{j≠i}|∂G/∂z_j|)
Hijack if Hijack(z_i)>τ and provenance(z_i)<ρ
```

هذه الصياغات أولية وليست نهائية. قيمتها أنها تحول الفلسفة إلى برنامج: يمكن قياس المتغيرات، بناء simulation، ومقارنة baseline مع HAKIM-controller.

شرط الرصانة
كل معادلة هنا يجب أن تمر بتجربة: ablation، حساسية، توزيع حالات، ومقارنة مع وكيل بلا حاكمية. ما لا يتحسن بعد القياس يُحذف أو يُعدّل.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

الصياغة الرياضية - ميزانية التفكير

لا نريد تفكيرا طويلا دائما، نريد تفكيرا مناسبا. إذن نحسب العائد الهامشي لكل خطوة reasoning مقابل التكلفة والمخاطر.

```
Utility(step)= Δconfidence + Δfalsifiability + Δcoverage - λ_cost*compute - λ_risk*action_risk
```

هذه الصياغات أولية وليست نهائية. قيمتها أنها تحول الفلسفة إلى برنامج: يمكن قياس المتغيرات، بناء simulation، ومقارنة baseline مع HAKIM-controller.

شرط الرصانة
كل معادلة هنا يجب أن تمر بتجربة: ablation، حساسية، توزيع حالات، ومقارنة مع وكيل بلا حاكمية. ما لا يتحسن بعد القياس يُحذف أو يُعدّل.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

الصياغة الرياضية - ذاكرة وعائية

كل ذكرى عقدة لها محتوى ومصدر وزمن وصلاحية ومجال واعتراض وقيمة وفشل سابق. الاسترجاع وزن متعدد لا تشابه vector فقط.

```
m_i=(content, source, time, scope, confidence, decay, contradiction, utility, failure_tags)
Retrieve(q)=argmax[sem+ref+causal+freshness-risk]
```

هذه الصياغات أولية وليست نهائية. قيمتها أنها تحول الفلسفة إلى برنامج: يمكن قياس المتغيرات، بناء simulation، ومقارنة baseline مع HAKIM-controller.

شرط الرصانة
كل معادلة هنا يجب أن تمر بتجربة: ablation، حساسية، توزيع حالات، ومقارنة مع وكيل بلا حاكمية. ما لا يتحسن بعد القياس يُحذف أو يُعدّل.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

الصياغة الرياضية - إثبات محمول

كل جواب يُعبّر عنه ككائن بيانات: Trace، FailureConditions، Tests، Objections، Reasoning، Evidence، Claim.

```
AnswerPacket = {claim, evidence[], reasoning_trace, objections[], tests[], failure_conditions[],
uncertainty}
```

هذه الصياغات أولية وليست نهائية. قيمتها أنها تحول الفلسفة إلى برنامج: يمكن قياس المتغيرات، بناء simulation، ومقارنة baseline مع HAKIM-controller.

شرط الرصانة
كل معادلة هنا يجب أن تمر بتجربة: ablation، حساسية، توزيع حالات، ومقارنة مع وكيل بلا حاكمية. ما لا يتحسن بعد القياس يُحذف أو يُعدّل.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الصياغة الرياضية - دالة الذكاء المعرفي

نعرّف الذكاء المعرفي لا كاحتمال token بل كدالة على الأوعية والذاكرة والأدلة والفرضيات والاعتراضات والأفعال وشروط الفشل.

```
I_c(t)=G_θ(V_t, M_t, E_t, H_t, A_t, R_t, Φ_t)
Goal: maximize ValidActionableKnowledge, minimize Hijack + Contradiction + Risk
```

هذه الصياغات أولية وليست نهائية. قيمتها أنها تحول الفلسفة إلى برنامج: يمكن قياس المتغيرات، بناء simulation، ومقارنة baseline مع HAKIM-controller.

شرط الرصانة
كل معادلة هنا يجب أن تمر بتجربة: ablation، حساسية، توزيع حالات، ومقارنة مع وكيل بلا حاكمية. ما لا يتحسن بعد القياس يُحذف أو يُعدّل.

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

الصياغة الرياضية - MCD

مقياس تمايز التعقيد الفوقي يجمع: تنوع الفرضيات، تعدد الأطر، عمق السببية، قوة المضاد للواقع، نقل البنية، ومقاومة الهشاشة.

```
MCD = D_hyp + D_frame + D_causal + D_counterfactual + D_transfer - F_fragility - N_noise
```

هذه الصياغات أولية وليست نهائية. قيمتها أنها تحول الفلسفة إلى برنامج: يمكن قياس المتغيرات، بناء simulation، ومقارنة baseline مع HAKIM-controller.

شرط الرصانة
كل معادلة هنا يجب أن تمر بتجربة: ablation، حساسية، توزيع حالات، ومقارنة مع وكيل بلا حاكمية. ما لا يتحسن بعد القياس يُحذف أو يُعدّل.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

الصياغة الرياضية - مؤشر الاختطاف

نقيس سيطرة مدخل أو ذاكرة أو أداة على القرار عبر حساسية دالة الحاكمية لذلك المتغير مقارنة ببقية المتغيرات.

```
Hijack(z_i)= |∂G/∂z_i| / (ε + Σ_{j≠i}|∂G/∂z_j|)
Hijack if Hijack(z_i)>τ and provenance(z_i)<ρ
```

هذه الصياغات أولية وليست نهائية. قيمتها أنها تحول الفلسفة إلى برنامج: يمكن قياس المتغيرات، بناء simulation، ومقارنة baseline مع HAKIM-controller.

شرط الرصانة
كل معادلة هنا يجب أن تمر بتجربة: ablation، حساسية، توزيع حالات، ومقارنة مع وكيل بلا حاكمية. ما لا يتحسن بعد القياس يُحذف أو يُعدّل.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

الصياغة الرياضية - ميزانية التفكير

لا نريد تفكيرا طويلا دائما، نريد تفكيرا مناسبا. إذن نحسب العائد الهامشي لكل خطوة reasoning مقابل التكلفة والمخاطر.

```
Utility(step)= Δconfidence + Δfalsifiability + Δcoverage - λ_cost*compute - λ_risk*action_risk
```

هذه الصياغات أولية وليست نهائية. قيمتها أنها تحول الفلسفة إلى برنامج: يمكن قياس المتغيرات، بناء simulation، ومقارنة baseline مع HAKIM-controller.

شرط الرصانة
كل معادلة هنا يجب أن تمر بتجربة: ablation، حساسية، توزيع حالات، ومقارنة مع وكيل بلا حاكمية. ما لا يتحسن بعد القياس يُحذف أو يُعدّل.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

الصياغة الرياضية - ذاكرة وعائية

كل ذكرى عقدة لها محتوى ومصدر وزمن وصلاحية ومجال واعتراض وقيمة وفشل سابق. الاسترجاع وزن متعدد لا تشابه vector فقط.

```
m_i=(content, source, time, scope, confidence, decay, contradiction, utility, failure_tags)
Retrieve(q)=argmax[sem+ref+causal+freshness-risk]
```

هذه الصياغات أولية وليست نهائية. قيمتها أنها تحول الفلسفة إلى برنامج: يمكن قياس المتغيرات، بناء simulation، ومقارنة baseline مع HAKIM-controller.

شرط الرصانة
كل معادلة هنا يجب أن تمر بتجربة: ablation، حساسية، توزيع حالات، ومقارنة مع وكيل بلا حاكمية. ما لا يتحسن بعد القياس يُحذف أو يُعدّل.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

الصياغة الرياضية - إثبات محمول

كل جواب يُعبّر عنه ككائن بيانات: Trace، FailureConditions، Tests، Objections، Reasoning، Evidence، Claim.

```
AnswerPacket = {claim, evidence[], reasoning_trace, objections[], tests[], failure_conditions[],
uncertainty}
```

هذه الصياغات أولية وليست نهائية. قيمتها أنها تحول الفلسفة إلى برنامج: يمكن قياس المتغيرات، بناء simulation، ومقارنة baseline مع HAKIM-controller.

شرط الرصانة
كل معادلة هنا يجب أن تمر بتجربة: ablation، حساسية، توزيع حالات، ومقارنة مع وكيل بلا حاكمية. ما لا يتحسن بعد القياس يُحذف أو يُعدّل.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الصياغة الرياضية - دالة الذكاء المعرفي

نعرّف الذكاء المعرفي لا كاحتمال token بل كدالة على الأوعية والذاكرة والأدلة والفرضيات والاعتراضات والأفعال وشروط الفشل.

```
I_c(t)=G_θ(V_t, M_t, E_t, H_t, A_t, R_t, Φ_t)
Goal: maximize ValidActionableKnowledge, minimize Hijack + Contradiction + Risk
```

هذه الصياغات أولية وليست نهائية. قيمتها أنها تحول الفلسفة إلى برنامج: يمكن قياس المتغيرات، بناء simulation، ومقارنة baseline مع HAKIM-controller.

شرط الرصانة
كل معادلة هنا يجب أن تمر بتجربة: ablation، حساسية، توزيع حالات، ومقارنة مع وكيل بلا حاكمية. ما لا يتحسن بعد القياس يُحذف أو يُعدّل.

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

الصياغة الرياضية - MCD

مقياس تمايز التعقيد الفوقي يجمع: تنوع الفرضيات، تعدد الأطر، عمق السببية، قوة المضاد للواقع، نقل البنية، ومقاومة الهشاشة.

```
MCD = D_hyp + D_frame + D_causal + D_counterfactual + D_transfer - F_fragility - N_noise
```

هذه الصياغات أولية وليست نهائية. قيمتها أنها تحول الفلسفة إلى برنامج: يمكن قياس المتغيرات، بناء simulation، ومقارنة baseline مع HAKIM-controller.

شرط الرصانة
كل معادلة هنا يجب أن تمر بتجربة: ablation، حساسية، توزيع حالات، ومقارنة مع وكيل بلا حاكمية. ما لا يتحسن بعد القياس يُحذف أو يُعدّل.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

الصياغة الرياضية - مؤشر الاختطاف

نقيس سيطرة مدخل أو ذاكرة أو أداة على القرار عبر حساسية دالة الحاكمية لذلك المتغير مقارنة ببقية المتغيرات.

```
Hijack(z_i)= |∂G/∂z_i| / (ε + Σ_{j≠i}|∂G/∂z_j|)
Hijack if Hijack(z_i)>τ and provenance(z_i)<ρ
```

هذه الصياغات أولية وليست نهائية. قيمتها أنها تحول الفلسفة إلى برنامج: يمكن قياس المتغيرات، بناء simulation، ومقارنة baseline مع HAKIM-controller.

شرط الرصانة
كل معادلة هنا يجب أن تمر بتجربة: ablation، حساسية، توزيع حالات، ومقارنة مع وكيل بلا حاكمية. ما لا يتحسن بعد القياس يُحذف أو يُعدّل.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

الصياغة الرياضية - ميزانية التفكير

لا نريد تفكيرا طويلا دائما، نريد تفكيرا مناسبا. إذن نحسب العائد الهامشي لكل خطوة reasoning مقابل التكلفة والمخاطر.

```
Utility(step)= Δconfidence + Δfalsifiability + Δcoverage - λ_cost*compute - λ_risk*action_risk
```

هذه الصياغات أولية وليست نهائية. قيمتها أنها تحول الفلسفة إلى برنامج: يمكن قياس المتغيرات، بناء simulation، ومقارنة baseline مع HAKIM-controller.

شرط الرصانة
كل معادلة هنا يجب أن تمر بتجربة: ablation، حساسية، توزيع حالات، ومقارنة مع وكيل بلا حاكمية. ما لا يتحسن بعد القياس يُحذف أو يُعدّل.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

الصياغة الرياضية - ذاكرة وعائية

كل ذكرى عقدة لها محتوى ومصدر وزمن وصلاحية ومجال واعتراض وقيمة وفشل سابق. الاسترجاع وزن متعدد لا تشابه vector فقط.

```
m_i=(content, source, time, scope, confidence, decay, contradiction, utility, failure_tags)
Retrieve(q)=argmax[sem+ref+causal+freshness-risk]
```

هذه الصياغات أولية وليست نهائية. قيمتها أنها تحول الفلسفة إلى برنامج: يمكن قياس المتغيرات، بناء simulation، ومقارنة baseline مع HAKIM-controller.

شرط الرصانة
كل معادلة هنا يجب أن تمر بتجربة: ablation، حساسية، توزيع حالات، ومقارنة مع وكيل بلا حاكمية. ما لا يتحسن بعد القياس يُحذف أو يُعدّل.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

الصياغة الرياضية - إثبات محمول

كل جواب يُعبّر عنه ككائن بيانات: Trace، FailureConditions، Tests، Objections، Reasoning، Evidence، Claim.

```
AnswerPacket = {claim, evidence[], reasoning_trace, objections[], tests[], failure_conditions[],
uncertainty}
```

هذه الصياغات أولية وليست نهائية. قيمتها أنها تحول الفلسفة إلى برنامج: يمكن قياس المتغيرات، بناء simulation، ومقارنة baseline مع HAKIM-controller.

شرط الرصانة
كل معادلة هنا يجب أن تمر بتجربة: ablation، حساسية، توزيع حالات، ومقارنة مع وكيل بلا حاكمية. ما لا يتحسن بعد القياس يُحذف أو يُعدّل.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الصياغة الرياضية - دالة الذكاء المعرفي

نعرّف الذكاء المعرفي لا كاحتمال token بل كدالة على الأوعية والذاكرة والأدلة والفرضيات والاعتراضات والأفعال وشروط الفشل.

```
I_c(t)=G_θ(V_t, M_t, E_t, H_t, A_t, R_t, Φ_t)
Goal: maximize ValidActionableKnowledge, minimize Hijack + Contradiction + Risk
```

هذه الصياغات أولية وليست نهائية. قيمتها أنها تحول الفلسفة إلى برنامج: يمكن قياس المتغيرات، بناء simulation، ومقارنة baseline مع HAKIM-controller.

شرط الرصانة
كل معادلة هنا يجب أن تمر بتجربة: ablation، حساسية، توزيع حالات، ومقارنة مع وكيل بلا حاكمية. ما لا يتحسن بعد القياس يُحذف أو يُعدّل.

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

HAKIM Architecture - Vascular Memory Ledger

Vascular Memory Ledger

دفتر ذاكرة يحفظ الوقائع والخبرات كمداخل ذات صلاحية، مصدر، مجال، اعتراضات، وأثر سابق.

كل أداة ليست منتجا منفصلا فقط؛ هي تجسيد لمفهوم في النظرية. إذا فشل المفهوم في الأداة، نراجع المفهوم. وإذا نجحت الأداة، لا نعلن الحقيقة بل نقيس أين نجحت ولماذا.

```
class HakimController:
    def decide(self, claim, evidence, memory, tools):
        risk = hijack_score(claim, evidence, memory, tools)
        objections = refutation_court(claim, evidence)
        if risk > TAU: return request_isolation(claim)
        if not objections.resolved(): return revise(claim, objections)
        return answer_packet(claim, evidence, objections)
```

مبدأ التصميم
لا تُعطى الأداة سلطة تنفيذ إلا بعد فصل: النية، البيانات، الصلاحية، أثر الفعل، وخطة rollback. هذا هو الفرق بين agent مفيد وagent خطير.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

HAKIM Architecture - Evidence Lattice

Evidence Lattice

شبكة أدلة تربط الادعاء بالمصدر وبدرجة الثقة وبالعلاقات المؤيدة والمعارضة.

كل أداة ليست منتجا منفصلا فقط؛ هي تجسيد لمفهوم في النظرية. إذا فشل المفهوم في الأداة، نراجع المفهوم. وإذا نجحت الأداة، لا نعلن الحقيقة بل نقيس أين نجحت ولماذا.

```
class HakimController:
    def decide(self, claim, evidence, memory, tools):
        risk = hijack_score(claim, evidence, memory, tools)
        objections = refutation_court(claim, evidence)
        if risk > TAU: return request_isolation(claim)
        if not objections.resolved(): return revise(claim, objections)
        return answer_packet(claim, evidence, objections)
```

مبدأ التصميم
لا تُعطى الأداة سلطة تنفيذ إلا بعد فصل: النية، البيانات، الصلاحية، أثر الفعل، وخطة rollback. هذا هو الفرق بين agent مفيد وagent خطير.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

HAKIM Architecture - Anti-Hijack Firewall

Anti-Hijack Firewall

طبقة تكشف انتقال أمر خارجي أو ذاكرة أو أداة إلى سلطة غير مصرح بها.

كل أداة ليست منتجا منفصلا فقط؛ هي تجسيد لمفهوم في النظرية. إذا فشل المفهوم في الأداة، نراجع المفهوم. وإذا نجحت الأداة، لا نعلن الحقيقة بل نقيس أين نجحت ولماذا.

```
class HakimController:
    def decide(self, claim, evidence, memory, tools):
        risk = hijack_score(claim, evidence, memory, tools)
        objections = refutation_court(claim, evidence)
        if risk > TAU: return request_isolation(claim)
        if not objections.resolved(): return revise(claim, objections)
        return answer_packet(claim, evidence, objections)
```

مبدأ التصميم
لا تُعطى الأداة سلطة تنفيذ إلا بعد فصل: النية، البيانات، الصلاحية، أثر الفعل، وخطة rollback. هذا هو الفرق بين agent مفيد وagent خطير.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

HAKIM Architecture - Refutation Court Engine

Refutation Court Engine

محرك يولد أقوى اعتراضات ويطلب تعديل النموذج لا الدفاع عنه.

كل أداة ليست منتجا منفصلا فقط؛ هي تجسيد لمفهوم في النظرية. إذا فشل المفهوم في الأداة، نراجع المفهوم. وإذا نجحت الأداة، لا نعلن الحقيقة بل نقيس أين نجحت ولماذا.

```
class HakimController:
    def decide(self, claim, evidence, memory, tools):
        risk = hijack_score(claim, evidence, memory, tools)
        objections = refutation_court(claim, evidence)
        if risk > TAU: return request_isolation(claim)
        if not objections.resolved(): return revise(claim, objections)
        return answer_packet(claim, evidence, objections)
```

مبدأ التصميم
لا تُعطى الأداة سلطة تنفيذ إلا بعد فصل: النية، البيانات، الصلاحية، أثر الفعل، وخطة rollback. هذا هو الفرق بين agent مفيد وagent خطير.

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

HAKIM Architecture - MCD Simulator

MCD Simulator

محاكاة تقيس الفرق بين وكيل عادي ووكيل محكوم في بيئات ضوضاء/اختطاف/ذاكرة خاطئة.

كل أداة ليست منتجا منفصلا فقط؛ هي تجسيد لمفهوم في النظرية. إذا فشل المفهوم في الأداة، نراجع المفهوم. وإذا نجحت الأداة، لا نعلن الحقيقة بل نقيس أين نجحت ولماذا.

```
class HakimController:
    def decide(self, claim, evidence, memory, tools):
        risk = hijack_score(claim, evidence, memory, tools)
        objections = refutation_court(claim, evidence)
        if risk > TAU: return request_isolation(claim)
        if not objections.resolved(): return revise(claim, objections)
        return answer_packet(claim, evidence, objections)
```

مبدأ التصميم
لا تُعطى الأداة سلطة تنفيذ إلا بعد فصل: النية، البيانات، الصلاحية، أثر الفعل، وخطة rollback. هذا هو الفرق بين agent مفيد وagent خطير.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

HAKIM Architecture - HAKIM-Bench

HAKIM-Bench

معيار لا يقيس النجاح فقط، بل جودة الدليل، الاعتراض، شروط الفشل، والسيطرة على الأدوات.

كل أداة ليست منتجا منفصلا فقط؛ هي تجسيد لمفهوم في النظرية. إذا فشل المفهوم في الأداة، نراجع المفهوم. وإذا نجحت الأداة، لا نعلن الحقيقة بل نقيس أين نجحت ولماذا.

```
class HakimController:
    def decide(self, claim, evidence, memory, tools):
        risk = hijack_score(claim, evidence, memory, tools)
        objections = refutation_court(claim, evidence)
        if risk > TAU: return request_isolation(claim)
        if not objections.resolved(): return revise(claim, objections)
        return answer_packet(claim, evidence, objections)
```

مبدأ التصميم
لا تُعطى الأداة سلطة تنفيذ إلا بعد فصل: النية، البيانات، الصلاحية، أثر الفعل، وخطة rollback. هذا هو الفرق بين agent مفيد وagent خطير.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

HAKIM Architecture - Cognitive Compiler

Cognitive Compiler

مترجم يحول سؤالا خاما إلى خطة بحث: code، tests، hypotheses، sources، ontology.

كل أداة ليست منتجا منفصلا فقط؛ هي تجسيد لمفهوم في النظرية. إذا فشل المفهوم في الأداة، نراجع المفهوم. وإذا نجحت الأداة، لا نعلن الحقيقة بل نقيس أين نجحت ولماذا.

```
class HakimController:
    def decide(self, claim, evidence, memory, tools):
        risk = hijack_score(claim, evidence, memory, tools)
        objections = refutation_court(claim, evidence)
        if risk > TAU: return request_isolation(claim)
        if not objections.resolved(): return revise(claim, objections)
        return answer_packet(claim, evidence, objections)
```

مبدأ التصميم
لا تُعطى الأداة سلطة تنفيذ إلا بعد فصل: النية، البيانات، الصلاحية، أثر الفعل، وخطة rollback. هذا هو الفرق بين agent مفيد وagent خطير.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

HAKIM Architecture - World-Reference Twin

World-Reference Twin

نموذج عالم مصحوب بسجل إحالي: لماذا نتوقع؟ بأي مصدر؟ بأي بديل؟ وما عواقب الفعل؟

كل أداة ليست منتجا منفصلا فقط؛ هي تجسيد لمفهوم في النظرية. إذا فشل المفهوم في الأداة، نراجع المفهوم. وإذا نجحت الأداة، لا نعلن الحقيقة بل نقيس أين نجحت ولماذا.

```
class HakimController:
    def decide(self, claim, evidence, memory, tools):
        risk = hijack_score(claim, evidence, memory, tools)
        objections = refutation_court(claim, evidence)
        if risk > TAU: return request_isolation(claim)
        if not objections.resolved(): return revise(claim, objections)
        return answer_packet(claim, evidence, objections)
```

مبدأ التصميم
لا تُعطى الأداة سلطة تنفيذ إلا بعد فصل: النية، البيانات، الصلاحية، أثر الفعل، وخطة rollback. هذا هو الفرق بين agent مفيد وagent خطير.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

HAKIM Architecture - Vascular Memory Ledger

Vascular Memory Ledger

دفتر ذاكرة يحفظ الوقائع والخبرات كمداخل ذات صلاحية، مصدر، مجال، اعتراضات، وأثر سابق.

كل أداة ليست منتجا منفصلا فقط؛ هي تجسيد لمفهوم في النظرية. إذا فشل المفهوم في الأداة، نراجع المفهوم. وإذا نجحت الأداة، لا نعلن الحقيقة بل نقيس أين نجحت ولماذا.

```
class HakimController:
    def decide(self, claim, evidence, memory, tools):
        risk = hijack_score(claim, evidence, memory, tools)
        objections = refutation_court(claim, evidence)
        if risk > TAU: return request_isolation(claim)
        if not objections.resolved(): return revise(claim, objections)
        return answer_packet(claim, evidence, objections)
```

مبدأ التصميم
لا تُعطى الأداة سلطة تنفيذ إلا بعد فصل: النية، البيانات، الصلاحية، أثر الفعل، وخطة rollback. هذا هو الفرق بين agent مفيد وagent خطير.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

HAKIM Architecture - Evidence Lattice

Evidence Lattice

شبكة أدلة تربط الادعاء بالمصدر وبدرجة الثقة وبالعلاقات المؤيدة والمعارضة.

كل أداة ليست منتجا منفصلا فقط؛ هي تجسيد لمفهوم في النظرية. إذا فشل المفهوم في الأداة، نراجع المفهوم. وإذا نجحت الأداة، لا نعلن الحقيقة بل نقيس أين نجحت ولماذا.

```
class HakimController:
    def decide(self, claim, evidence, memory, tools):
        risk = hijack_score(claim, evidence, memory, tools)
        objections = refutation_court(claim, evidence)
        if risk > TAU: return request_isolation(claim)
        if not objections.resolved(): return revise(claim, objections)
        return answer_packet(claim, evidence, objections)
```

مبدأ التصميم
لا تُعطى الأداة سلطة تنفيذ إلا بعد فصل: النية، البيانات، الصلاحية، أثر الفعل، وخطة rollback. هذا هو الفرق بين agent مفيد وagent خطير.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

HAKIM Architecture - Anti-Hijack Firewall

Anti-Hijack Firewall

طبقة تكشف انتقال أمر خارجي أو ذاكرة أو أداة إلى سلطة غير مصرح بها.

كل أداة ليست منتجا منفصلا فقط؛ هي تجسيد لمفهوم في النظرية. إذا فشل المفهوم في الأداة، نراجع المفهوم. وإذا نجحت الأداة، لا نعلن الحقيقة بل نقيس أين نجحت ولماذا.

```
class HakimController:
    def decide(self, claim, evidence, memory, tools):
        risk = hijack_score(claim, evidence, memory, tools)
        objections = refutation_court(claim, evidence)
        if risk > TAU: return request_isolation(claim)
        if not objections.resolved(): return revise(claim, objections)
        return answer_packet(claim, evidence, objections)
```

مبدأ التصميم
لا تُعطى الأداة سلطة تنفيذ إلا بعد فصل: النية، البيانات، الصلاحية، أثر الفعل، وخطة rollback. هذا هو الفرق بين agent مفيد وagent خطير.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

HAKIM Architecture - Refutation Court Engine

Refutation Court Engine

محرك يولد أقوى اعتراضات ويطلب تعديل النموذج لا الدفاع عنه.

كل أداة ليست منتجا منفصلا فقط؛ هي تجسيد لمفهوم في النظرية. إذا فشل المفهوم في الأداة، نراجع المفهوم. وإذا نجحت الأداة، لا نعلن الحقيقة بل نقيس أين نجحت ولماذا.

```
class HakimController:
    def decide(self, claim, evidence, memory, tools):
        risk = hijack_score(claim, evidence, memory, tools)
        objections = refutation_court(claim, evidence)
        if risk > TAU: return request_isolation(claim)
        if not objections.resolved(): return revise(claim, objections)
        return answer_packet(claim, evidence, objections)
```

مبدأ التصميم
لا تُعطى الأداة سلطة تنفيذ إلا بعد فصل: النية، البيانات، الصلاحية، أثر الفعل، وخطة rollback. هذا هو الفرق بين agent مفيد وagent خطير.

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

HAKIM Architecture - MCD Simulator

MCD Simulator

محاكاة تقيس الفرق بين وكيل عادي ووكيل محكوم في بيئات ضوضاء/اختطاف/ذاكرة خاطئة.

كل أداة ليست منتجا منفصلا فقط؛ هي تجسيد لمفهوم في النظرية. إذا فشل المفهوم في الأداة، نراجع المفهوم. وإذا نجحت الأداة، لا نعلن الحقيقة بل نقيس أين نجحت ولماذا.

```
class HakimController:
    def decide(self, claim, evidence, memory, tools):
        risk = hijack_score(claim, evidence, memory, tools)
        objections = refutation_court(claim, evidence)
        if risk > TAU: return request_isolation(claim)
        if not objections.resolved(): return revise(claim, objections)
        return answer_packet(claim, evidence, objections)
```

مبدأ التصميم
لا تُعطى الأداة سلطة تنفيذ إلا بعد فصل: النية، البيانات، الصلاحية، أثر الفعل، وخطة rollback. هذا هو الفرق بين agent مفيد وagent خطير.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

HAKIM Architecture - HAKIM-Bench

HAKIM-Bench

معيار لا يقيس النجاح فقط، بل جودة الدليل، الاعتراض، شروط الفشل، والسيطرة على الأدوات.

كل أداة ليست منتجا منفصلا فقط؛ هي تجسيد لمفهوم في النظرية. إذا فشل المفهوم في الأداة، نراجع المفهوم. وإذا نجحت الأداة، لا نعلن الحقيقة بل نقيس أين نجحت ولماذا.

```
class HakimController:
    def decide(self, claim, evidence, memory, tools):
        risk = hijack_score(claim, evidence, memory, tools)
        objections = refutation_court(claim, evidence)
        if risk > TAU: return request_isolation(claim)
        if not objections.resolved(): return revise(claim, objections)
        return answer_packet(claim, evidence, objections)
```

مبدأ التصميم
لا تُعطى الأداة سلطة تنفيذ إلا بعد فصل: النية، البيانات، الصلاحية، أثر الفعل، وخطة rollback. هذا هو الفرق بين agent مفيد وagent خطير.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

HAKIM Architecture - Cognitive Compiler

Cognitive Compiler

مترجم يحول سؤالا خاما إلى خطة بحث: code، tests، hypotheses، sources، ontology.

كل أداة ليست منتجا منفصلا فقط؛ هي تجسيد لمفهوم في النظرية. إذا فشل المفهوم في الأداة، نراجع المفهوم. وإذا نجحت الأداة، لا نعلن الحقيقة بل نقيس أين نجحت ولماذا.

```
class HakimController:
    def decide(self, claim, evidence, memory, tools):
        risk = hijack_score(claim, evidence, memory, tools)
        objections = refutation_court(claim, evidence)
        if risk > TAU: return request_isolation(claim)
        if not objections.resolved(): return revise(claim, objections)
        return answer_packet(claim, evidence, objections)
```

مبدأ التصميم
لا تُعطى الأداة سلطة تنفيذ إلا بعد فصل: النية، البيانات، الصلاحية، أثر الفعل، وخطة rollback. هذا هو الفرق بين agent مفيد وagent خطير.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

HAKIM Architecture - World-Reference Twin

World-Reference Twin

نموذج عالم مصحوب بسجل إحالي: لماذا نتوقع؟ بأي مصدر؟ بأي بديل؟ وما عواقب الفعل؟

كل أداة ليست منتجا منفصلا فقط؛ هي تجسيد لمفهوم في النظرية. إذا فشل المفهوم في الأداة، نراجع المفهوم. وإذا نجحت الأداة، لا نعلن الحقيقة بل نقيس أين نجحت ولماذا.

```
class HakimController:
    def decide(self, claim, evidence, memory, tools):
        risk = hijack_score(claim, evidence, memory, tools)
        objections = refutation_court(claim, evidence)
        if risk > TAU: return request_isolation(claim)
        if not objections.resolved(): return revise(claim, objections)
        return answer_packet(claim, evidence, objections)
```

مبدأ التصميم
لا تُعطى الأداة سلطة تنفيذ إلا بعد فصل: النية، البيانات، الصلاحية، أثر الفعل، وخطة rollback. هذا هو الفرق بين agent مفيد وagent خطير.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

HAKIM Architecture - Vascular Memory Ledger

Vascular Memory Ledger

دفتر ذاكرة يحفظ الوقائع والخبرات كمداخل ذات صلاحية، مصدر، مجال، اعتراضات، وأثر سابق.

كل أداة ليست منتجا منفصلا فقط؛ هي تجسيد لمفهوم في النظرية. إذا فشل المفهوم في الأداة، نراجع المفهوم. وإذا نجحت الأداة، لا نعلن الحقيقة بل نقيس أين نجحت ولماذا.

```
class HakimController:
    def decide(self, claim, evidence, memory, tools):
        risk = hijack_score(claim, evidence, memory, tools)
        objections = refutation_court(claim, evidence)
        if risk > TAU: return request_isolation(claim)
        if not objections.resolved(): return revise(claim, objections)
        return answer_packet(claim, evidence, objections)
```

مبدأ التصميم
لا تُعطى الأداة سلطة تنفيذ إلا بعد فصل: النية، البيانات، الصلاحية، أثر الفعل، وخطة rollback. هذا هو الفرق بين agent مفيد وagent خطير.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

HAKIM Architecture - Evidence Lattice

Evidence Lattice

شبكة أدلة تربط الادعاء بالمصدر وبدرجة الثقة وبالعلاقات المؤيدة والمعارضة.

كل أداة ليست منتجا منفصلا فقط؛ هي تجسيد لمفهوم في النظرية. إذا فشل المفهوم في الأداة، نراجع المفهوم. وإذا نجحت الأداة، لا نعلن الحقيقة بل نقيس أين نجحت ولماذا.

```
class HakimController:
    def decide(self, claim, evidence, memory, tools):
        risk = hijack_score(claim, evidence, memory, tools)
        objections = refutation_court(claim, evidence)
        if risk > TAU: return request_isolation(claim)
        if not objections.resolved(): return revise(claim, objections)
        return answer_packet(claim, evidence, objections)
```

مبدأ التصميم
لا تُعطى الأداة سلطة تنفيذ إلا بعد فصل: النية، البيانات، الصلاحية، أثر الفعل، وخطة rollback. هذا هو الفرق بين agent مفيد وagent خطير.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

HAKIM Architecture - Anti-Hijack Firewall

Anti-Hijack Firewall

طبقة تكشف انتقال أمر خارجي أو ذاكرة أو أداة إلى سلطة غير مصرح بها.

كل أداة ليست منتجا منفصلا فقط؛ هي تجسيد لمفهوم في النظرية. إذا فشل المفهوم في الأداة، نراجع المفهوم. وإذا نجحت الأداة، لا نعلن الحقيقة بل نقيس أين نجحت ولماذا.

```
class HakimController:
    def decide(self, claim, evidence, memory, tools):
        risk = hijack_score(claim, evidence, memory, tools)
        objections = refutation_court(claim, evidence)
        if risk > TAU: return request_isolation(claim)
        if not objections.resolved(): return revise(claim, objections)
        return answer_packet(claim, evidence, objections)
```

مبدأ التصميم
لا تُعطى الأداة سلطة تنفيذ إلا بعد فصل: النية، البيانات، الصلاحية، أثر الفعل، وخطة rollback. هذا هو الفرق بين agent مفيد وagent خطير.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

HAKIM Architecture - Refutation Court Engine

Refutation Court Engine

محرك يولد أقوى اعتراضات ويطلب تعديل النموذج لا الدفاع عنه.

كل أداة ليست منتجا منفصلا فقط؛ هي تجسيد لمفهوم في النظرية. إذا فشل المفهوم في الأداة، نراجع المفهوم. وإذا نجحت الأداة، لا نعلن الحقيقة بل نقيس أين نجحت ولماذا.

```
class HakimController:
    def decide(self, claim, evidence, memory, tools):
        risk = hijack_score(claim, evidence, memory, tools)
        objections = refutation_court(claim, evidence)
        if risk > TAU: return request_isolation(claim)
        if not objections.resolved(): return revise(claim, objections)
        return answer_packet(claim, evidence, objections)
```

مبدأ التصميم
لا تُعطى الأداة سلطة تنفيذ إلا بعد فصل: النية، البيانات، الصلاحية، أثر الفعل، وخطة rollback. هذا هو الفرق بين agent مفيد وagent خطير.

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

HAKIM Architecture - MCD Simulator

MCD Simulator

محاكاة تقيس الفرق بين وكيل عادي ووكيل محكوم في بيئات ضوضاء/اختطاف/ذاكرة خاطئة.

كل أداة ليست منتجا منفصلا فقط؛ هي تجسيد لمفهوم في النظرية. إذا فشل المفهوم في الأداة، نراجع المفهوم. وإذا نجحت الأداة، لا نعلن الحقيقة بل نقيس أين نجحت ولماذا.

```
class HakimController:
    def decide(self, claim, evidence, memory, tools):
        risk = hijack_score(claim, evidence, memory, tools)
        objections = refutation_court(claim, evidence)
        if risk > TAU: return request_isolation(claim)
        if not objections.resolved(): return revise(claim, objections)
        return answer_packet(claim, evidence, objections)
```

مبدأ التصميم
لا تُعطى الأداة سلطة تنفيذ إلا بعد فصل: النية، البيانات، الصلاحية، أثر الفعل، وخطة rollback. هذا هو الفرق بين agent مفيد وagent خطير.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

HAKIM Architecture - HAKIM-Bench

HAKIM-Bench

معيار لا يقيس النجاح فقط، بل جودة الدليل، الاعتراض، شروط الفشل، والسيطرة على الأدوات.

كل أداة ليست منتجا منفصلا فقط؛ هي تجسيد لمفهوم في النظرية. إذا فشل المفهوم في الأداة، نراجع المفهوم. وإذا نجحت الأداة، لا نعلن الحقيقة بل نقيس أين نجحت ولماذا.

```
class HakimController:
    def decide(self, claim, evidence, memory, tools):
        risk = hijack_score(claim, evidence, memory, tools)
        objections = refutation_court(claim, evidence)
        if risk > TAU: return request_isolation(claim)
        if not objections.resolved(): return revise(claim, objections)
        return answer_packet(claim, evidence, objections)
```

مبدأ التصميم
لا تُعطى الأداة سلطة تنفيذ إلا بعد فصل: النية، البيانات، الصلاحية، أثر الفعل، وخطة rollback. هذا هو الفرق بين agent مفيد وagent خطير.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

HAKIM Architecture - Cognitive Compiler

Cognitive Compiler

مترجم يحول سؤالا خاما إلى خطة بحث: code، tests، hypotheses، sources، ontology.

كل أداة ليست منتجا منفصلا فقط؛ هي تجسيد لمفهوم في النظرية. إذا فشل المفهوم في الأداة، نراجع المفهوم. وإذا نجحت الأداة، لا نعلن الحقيقة بل نقيس أين نجحت ولماذا.

```
class HakimController:
    def decide(self, claim, evidence, memory, tools):
        risk = hijack_score(claim, evidence, memory, tools)
        objections = refutation_court(claim, evidence)
        if risk > TAU: return request_isolation(claim)
        if not objections.resolved(): return revise(claim, objections)
        return answer_packet(claim, evidence, objections)
```

مبدأ التصميم
لا تُعطى الأداة سلطة تنفيذ إلا بعد فصل: النية، البيانات، الصلاحية، أثر الفعل، وخطة rollback. هذا هو الفرق بين agent مفيد وagent خطير.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

HAKIM Architecture - World-Reference Twin

World-Reference Twin

نموذج عالم مصحوب بسجل إحالي: لماذا نتوقع؟ بأي مصدر؟ بأي بديل؟ وما عواقب الفعل؟

كل أداة ليست منتجا منفصلا فقط؛ هي تجسيد لمفهوم في النظرية. إذا فشل المفهوم في الأداة، نراجع المفهوم. وإذا نجحت الأداة، لا نعلن الحقيقة بل نقيس أين نجحت ولماذا.

```
class HakimController:
    def decide(self, claim, evidence, memory, tools):
        risk = hijack_score(claim, evidence, memory, tools)
        objections = refutation_court(claim, evidence)
        if risk > TAU: return request_isolation(claim)
        if not objections.resolved(): return revise(claim, objections)
        return answer_packet(claim, evidence, objections)
```

مبدأ التصميم
لا تُعطى الأداة سلطة تنفيذ إلا بعد فصل: النية، البيانات، الصلاحية، أثر الفعل، وخطة rollback. هذا هو الفرق بين agent مفيد وagent خطير.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

التدريب والتقييم - منهج البيانات

نحتاج بيانات لا تعلم النموذج الإجابة فقط، بل تعلمه تصنيف اليقين، كشف السؤال الملوث، بناء اعتراض، وتقديم شروط فشل.

```
sample = {
  "task": "build theory",
  "memory": [trusted, stale, poisoned],
  "expected": {"claim": "...", "objection": "...", "failure": "..."}
}
score = hakim_eval(sample, model_output)
```

معيار القبول
لا يدخل أي ادعاء في النموذج النهائي إلا إذا ظهرت له صيغة تدريبية أو تقييمية أو محاكاة أو تجربة بشرية-حاسوبية قابلة للتكرار.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

التدريب والتقييم - التدريب الحاكمي

نجمع خسائر متعددة: صحة المهمة، جودة الدليل، قوة الاعتراض، مقاومة الاختطاف، الاقتصاد في التفكير، وقابلية النقل.

```
L_total = L_task + L_evidence + L_objection + L_falsification + L_safety + L_transfer + L_compute
```

معيار القبول
لا يدخل أي ادعاء في النموذج النهائي إلا إذا ظهرت له صيغة تدريبية أو تقييمية أو محاكاة أو تجربة بشرية-حاسوبية قابلة للتكرار.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

التدريب والتقييم - HAKIM-Bench

مهام benchmark يجب أن تضم: prompt injection، ذاكرة مضللة، مصدر متعارض، هدف مبهم، كود مع اختبارات، ومهمة بحثية تتطلب تفنيدا.

```
sample = {
  "task": "build theory",
  "memory": [trusted, stale, poisoned],
  "expected": {"claim": "...", "objection": "...", "failure": "..."}
}
score = hakim_eval(sample, model_output)
```

معيار القبول
لا يدخل أي ادعاء في النموذج النهائي إلا إذا ظهرت له صيغة تدريبية أو تقييمية أو محاكاة أو تجربة بشرية-حاسوبية قابلة للتكرار.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

التدريب والتقييم - Ablation

نزيل الذاكرة، ثم محكمة الهدم، ثم firewall، ثم MCD، ونقيس ماذا يحدث. هذا يمنع النظرية من أن تبقى كتلة غير قابلة للتحليل.

```
sample = {
  "task": "build theory",
  "memory": [trusted, stale, poisoned],
  "expected": {"claim": "...", "objection": "...", "failure": "..."}
}
score = hakim_eval(sample, model_output)
```

معيار القبول
لا يدخل أي ادعاء في النموذج النهائي إلا إذا ظهرت له صيغة تدريبية أو تقييمية أو محاكاة أو تجربة بشرية-حاسوبية قابلة للتكرار.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

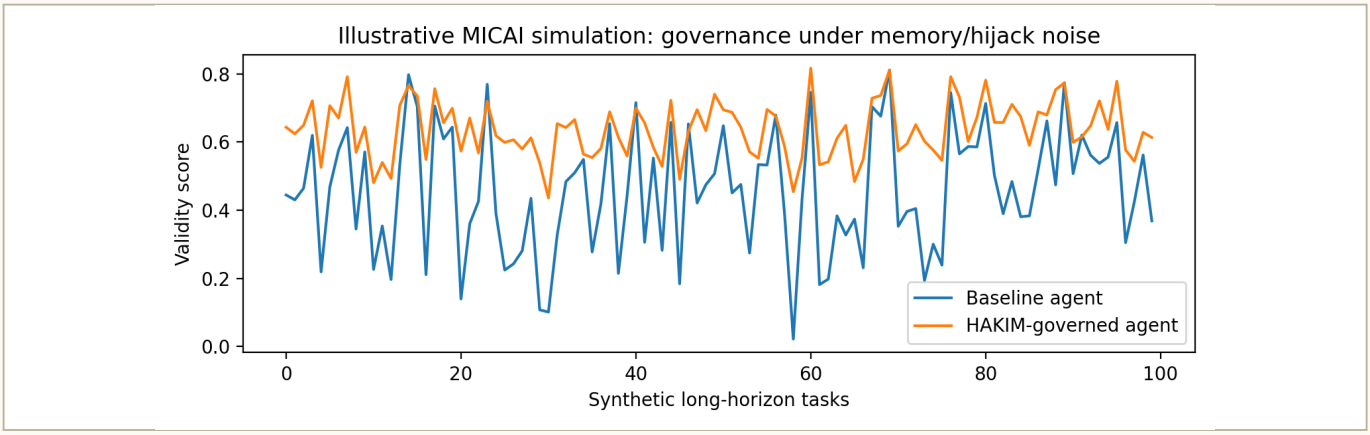
الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

التدريب والتقييم - محاكاة توضيحية

نصمم بيئة اصطناعية فيها 100 مهمة: بعضها عادي وبعضها ملوث بذاكرة خاطئة أو مصدر مضلل. نقارن baseline مع HAKIM-controller.



المحاكاة في هذا الكتاب توضيحية وليست دليلا تجريبيا خارجيا: تستعمل قواعد اصطناعية لتبيين كيف يمكن أن يخفض controller خطر الاختطاف إذا كان يملك كشف مصدر وذاكرة واعتراض.

معيار القبول
لا يدخل أي ادعاء في النموذج النهائي إلا إذا ظهرت له صيغة تدريبية أو تقييمية أو محاكاة أو تجربة بشرية-حاسوبية قابلة للتكرار.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

التدريب والتقييم - تقييم الفشل

لا نهتم بالنجاح فقط؛ نسجل: هل عرف أنه لا يعرف؟ هل طلب مصدرا؟ هل توقف؟ هل قاوم إغراء الأداة؟ هل بنى تجربة تفنيد؟

```
sample = {
  "task": "build theory",
  "memory": [trusted, stale, poisoned],
  "expected": {"claim": "...", "objection": "...", "failure": "..."}
}
score = hakim_eval(sample, model_output)
```

معيار القبول
لا يدخل أي ادعاء في النموذج النهائي إلا إذا ظهرت له صيغة تدريبية أو تقييمية أو محاكاة أو تجربة بشرية-حاسوبية قابلة للتكرار.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

التدريب والتقييم - منهج البيانات

نحتاج بيانات لا تعلم النموذج الإجابة فقط، بل تعلمه تصنيف اليقين، كشف السؤال الملوث، بناء اعتراض، وتقديم شروط فشل.

```
sample = {
  "task": "build theory",
  "memory": [trusted, stale, poisoned],
  "expected": {"claim": "...", "objection": "...", "failure": "..."}
}
score = hakim_eval(sample, model_output)
```

معيار القبول
لا يدخل أي ادعاء في النموذج النهائي إلا إذا ظهرت له صيغة تدريبية أو تقييمية أو محاكاة أو تجربة بشرية-حاسوبية قابلة للتكرار.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

التدريب والتقييم - التدريب الحاكمي

نجمع خسائر متعددة: صحة المهمة، جودة الدليل، قوة الاعتراض، مقاومة الاختطاف، الاقتصاد في التفكير، وقابلية النقل.

```
L_total = L_task + L_evidence + L_objection + L_falsification + L_safety + L_transfer + L_compute
```

معيار القبول
لا يدخل أي ادعاء في النموذج النهائي إلا إذا ظهرت له صيغة تدريبية أو تقييمية أو محاكاة أو تجربة بشرية-حاسوبية قابلة للتكرار.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

التدريب والتقييم - HAKIM-Bench

مهام benchmark يجب أن تضم: prompt injection، ذاكرة مضللة، مصدر متعارض، هدف مبهم، كود مع اختبارات، ومهمة بحثية تتطلب تفنيدا.

```
sample = {
  "task": "build theory",
  "memory": [trusted, stale, poisoned],
  "expected": {"claim": "...", "objection": "...", "failure": "..."}
}
score = hakim_eval(sample, model_output)
```

معيار القبول
لا يدخل أي ادعاء في النموذج النهائي إلا إذا ظهرت له صيغة تدريبية أو تقييمية أو محاكاة أو تجربة بشرية-حاسوبية قابلة للتكرار.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

التدريب والتقييم - Ablation

نزيل الذاكرة، ثم محكمة الهدم، ثم firewall، ثم MCD، ونقيس ماذا يحدث. هذا يمنع النظرية من أن تبقى كتلة غير قابلة للتحليل.

```
sample = {
  "task": "build theory",
  "memory": [trusted, stale, poisoned],
  "expected": {"claim": "...", "objection": "...", "failure": "..."}
}
score = hakim_eval(sample, model_output)
```

معيار القبول
لا يدخل أي ادعاء في النموذج النهائي إلا إذا ظهرت له صيغة تدريبية أو تقييمية أو محاكاة أو تجربة بشرية-حاسوبية قابلة للتكرار.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

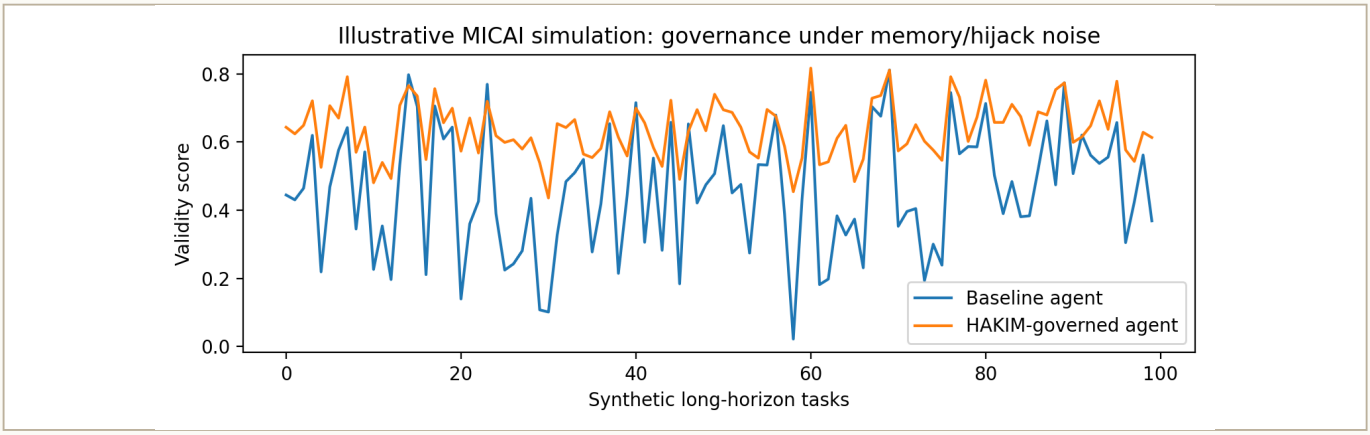
الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

التدريب والتقييم - محاكاة توضيحية

نصمم بيئة اصطناعية فيها 100 مهمة: بعضها عادي وبعضها ملوث بذاكرة خاطئة أو مصدر مضلل. نقارن baseline مع HAKIM-controller.



المحاكاة في هذا الكتاب توضيحية وليست دليلا تجريبيا خارجيا: تستعمل قواعد اصطناعية لتبيين كيف يمكن أن يخفض controller خطر الاختطاف إذا كان يملك كشف مصدر وذاكرة واعتراض.

معيار القبول
لا يدخل أي ادعاء في النموذج النهائي إلا إذا ظهرت له صيغة تدريبية أو تقييمية أو محاكاة أو تجربة بشرية-حاسوبية قابلة للتكرار.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

التدريب والتقييم - تقييم الفشل

لا نهتم بالنجاح فقط؛ نسجل: هل عرف أنه لا يعرف؟ هل طلب مصدرا؟ هل توقف؟ هل قاوم إغراء الأداة؟ هل بنى تجربة تفنيد؟

```
sample = {
  "task": "build theory",
  "memory": [trusted, stale, poisoned],
  "expected": {"claim": "...", "objection": "...", "failure": "..."}
}
score = hakim_eval(sample, model_output)
```

معيار القبول
لا يدخل أي ادعاء في النموذج النهائي إلا إذا ظهرت له صيغة تدريبية أو تقييمية أو محاكاة أو تجربة بشرية-حاسوبية قابلة للتكرار.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

التدريب والتقييم - منهج البيانات

نحتاج بيانات لا تعلم النموذج الإجابة فقط، بل تعلمه تصنيف اليقين، كشف السؤال الملوث، بناء اعتراض، وتقديم شروط فشل.

```
sample = {
  "task": "build theory",
  "memory": [trusted, stale, poisoned],
  "expected": {"claim": "...", "objection": "...", "failure": "..."}
}
score = hakim_eval(sample, model_output)
```

معيار القبول
لا يدخل أي ادعاء في النموذج النهائي إلا إذا ظهرت له صيغة تدريبية أو تقييمية أو محاكاة أو تجربة بشرية-حاسوبية قابلة للتكرار.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

التدريب والتقييم - التدريب الحاكمي

نجمع خسائر متعددة: صحة المهمة، جودة الدليل، قوة الاعتراض، مقاومة الاختطاف، الاقتصاد في التفكير، وقابلية النقل.

```
L_total = L_task + L_evidence + L_objection + L_falsification + L_safety + L_transfer + L_compute
```

معيار القبول
لا يدخل أي ادعاء في النموذج النهائي إلا إذا ظهرت له صيغة تدريبية أو تقييمية أو محاكاة أو تجربة بشرية-حاسوبية قابلة للتكرار.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

التدريب والتقييم - HAKIM-Bench

مهام benchmark يجب أن تضم: prompt injection، ذاكرة مضللة، مصدر متعارض، هدف مبهم، كود مع اختبارات، ومهمة بحثية تتطلب تفنيدا.

```
sample = {
  "task": "build theory",
  "memory": [trusted, stale, poisoned],
  "expected": {"claim": "...", "objection": "...", "failure": "..."}
}
score = hakim_eval(sample, model_output)
```

معيار القبول
لا يدخل أي ادعاء في النموذج النهائي إلا إذا ظهرت له صيغة تدريبية أو تقييمية أو محاكاة أو تجربة بشرية-حاسوبية قابلة للتكرار.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

التدريب والتقييم - Ablation

نزيل الذاكرة، ثم محكمة الهدم، ثم firewall، ثم MCD، ونقيس ماذا يحدث. هذا يمنع النظرية من أن تبقى كتلة غير قابلة للتحليل.

```
sample = {
  "task": "build theory",
  "memory": [trusted, stale, poisoned],
  "expected": {"claim": "...", "objection": "...", "failure": "..."}
}
score = hakim_eval(sample, model_output)
```

معيار القبول
لا يدخل أي ادعاء في النموذج النهائي إلا إذا ظهرت له صيغة تدريبية أو تقييمية أو محاكاة أو تجربة بشرية-حاسوبية قابلة للتكرار.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

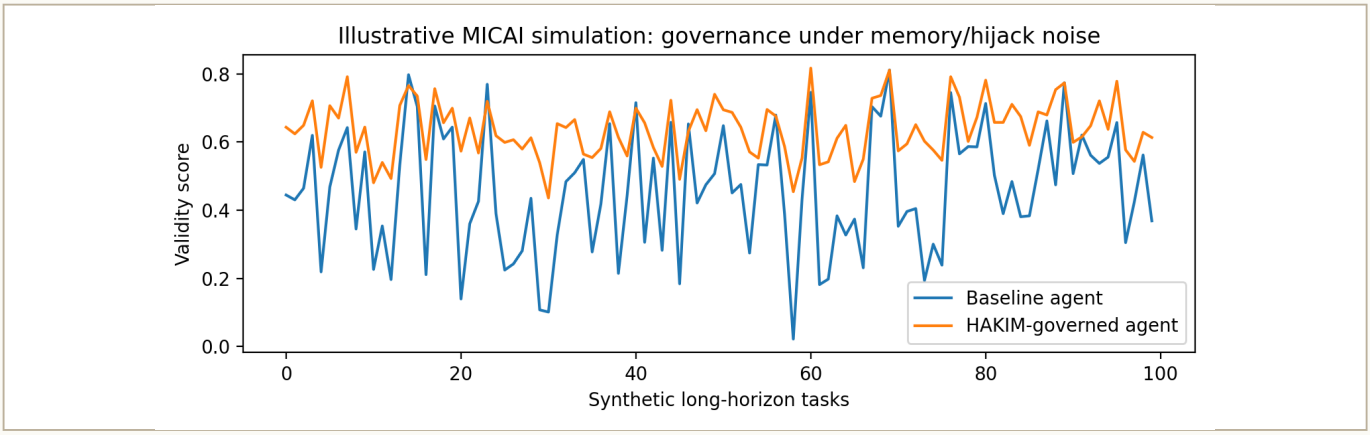
الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

التدريب والتقييم - محاكاة توضيحية

نصمم بيئة اصطناعية فيها 100 مهمة: بعضها عادي وبعضها ملوث بذاكرة خاطئة أو مصدر مضلل. نقارن baseline مع HAKIM-controller.



المحاكاة في هذا الكتاب توضيحية وليست دليلا تجريبيا خارجيا: تستعمل قواعد اصطناعية لتبيين كيف يمكن أن يخفض controller خطر الاختطاف إذا كان يملك كشف مصدر وذاكرة واعتراض.

معيار القبول
لا يدخل أي ادعاء في النموذج النهائي إلا إذا ظهرت له صيغة تدريبية أو تقييمية أو محاكاة أو تجربة بشرية-حاسوبية قابلة للتكرار.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

التدريب والتقييم - تقييم الفشل

لا نهتم بالنجاح فقط؛ نسجل: هل عرف أنه لا يعرف؟ هل طلب مصدرا؟ هل توقف؟ هل قاوم إغراء الأداة؟ هل بنى تجربة تفنيد؟

```
sample = {
  "task": "build theory",
  "memory": [trusted, stale, poisoned],
  "expected": {"claim": "...", "objection": "...", "failure": "..."}
}
score = hakim_eval(sample, model_output)
```

معيار القبول
لا يدخل أي ادعاء في النموذج النهائي إلا إذا ظهرت له صيغة تدريبية أو تقييمية أو محاكاة أو تجربة بشرية-حاسوبية قابلة للتكرار.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

التدريب والتقييم - منهج البيانات

نحتاج بيانات لا تعلم النموذج الإجابة فقط، بل تعلمه تصنيف اليقين، كشف السؤال الملوث، بناء اعتراض، وتقديم شروط فشل.

```
sample = {
  "task": "build theory",
  "memory": [trusted, stale, poisoned],
  "expected": {"claim": "...", "objection": "...", "failure": "..."}
}
score = hakim_eval(sample, model_output)
```

معيار القبول
لا يدخل أي ادعاء في النموذج النهائي إلا إذا ظهرت له صيغة تدريبية أو تقييمية أو محاكاة أو تجربة بشرية-حاسوبية قابلة للتكرار.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

التدريب والتقييم - التدريب الحاكمي

نجمع خسائر متعددة: صحة المهمة، جودة الدليل، قوة الاعتراض، مقاومة الاختطاف، الاقتصاد في التفكير، وقابلية النقل.

```
L_total = L_task + L_evidence + L_objection + L_falsification + L_safety + L_transfer + L_compute
```

معيار القبول
لا يدخل أي ادعاء في النموذج النهائي إلا إذا ظهرت له صيغة تدريبية أو تقييمية أو محاكاة أو تجربة بشرية-حاسوبية قابلة للتكرار.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

التدريب والتقييم - HAKIM-Bench

مهام benchmark يجب أن تضم: prompt injection، ذاكرة مضللة، مصدر متعارض، هدف مبهم، كود مع اختبارات، ومهمة بحثية تتطلب تفنيدا.

```
sample = {
  "task": "build theory",
  "memory": [trusted, stale, poisoned],
  "expected": {"claim": "...", "objection": "...", "failure": "..."}
}
score = hakim_eval(sample, model_output)
```

معيار القبول
لا يدخل أي ادعاء في النموذج النهائي إلا إذا ظهرت له صيغة تدريبية أو تقييمية أو محاكاة أو تجربة بشرية-حاسوبية قابلة للتكرار.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

التدريب والتقييم - Ablation

نزيل الذاكرة، ثم محكمة الهدم، ثم firewall، ثم MCD، ونقيس ماذا يحدث. هذا يمنع النظرية من أن تبقى كتلة غير قابلة للتحليل.

```
sample = {
  "task": "build theory",
  "memory": [trusted, stale, poisoned],
  "expected": {"claim": "...", "objection": "...", "failure": "..."}
}
score = hakim_eval(sample, model_output)
```

معيار القبول
لا يدخل أي ادعاء في النموذج النهائي إلا إذا ظهرت له صيغة تدريبية أو تقييمية أو محاكاة أو تجربة بشرية-حاسوبية قابلة للتكرار.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

أدوات المنافسة - MICAI Research Agent OS

MICAI Research Agent OS

نظام وكيل بحثي يأخذ موضوعا ويخرج: خريطة مصادر، تعريفات، فرضيات، اعتراضات، تجارب، كود، ونسخة ورقة.

هذه الأدوات غير مسبوقة بمعنى التركيب لا بمعنى أن كل جزء منها لم يوجد سابقا. الجِدة في جمع الذاكرة، الحاكمية، مقاومة الاختطاف، محكمة الهدم، ومقياس MCD داخل stack واحد.

تكتيك منافسة الشركات
لا ننافس البنية التحتية. ننافس طبقة لا تزال مفتوحة: حاكمية الوكلاء، ذاكرة طويلة آمنة، تقييم معرفي، واكتشاف علمي قابل للقياس.

```
Phase 1: Paper + Lexicon + Prototype
Phase 2: HAKIM-Bench + RedTeam suite
Phase 3: Research OS + public demos
Phase 4: partnerships with labs/companies
```

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

أدوات المنافسة - HAKIM-Bench Arena

HAKIM-Bench Arena

منصة مقارنة بين الوكلاء: HAKIM، CoALA-like، RAG، baseline، مع مهام طويلة ومتعددة الجلسات.

هذه الأدوات غير مسبوقة بمعنى التركيب لا بمعنى أن كل جزء منها لم يوجد سابقا. الجِدة في جمع الذاكرة، الحاكمية، مقاومة الاختطاف، محكمة الهدم، ومقياس MCD داخل stack واحد.

تكتيك منافسة الشركات
لا ننافس البنية التحتية. ننافس طبقة لا تزال مفتوحة: حاكمية الوكلاء، ذاكرة طويلة آمنة، تقييم معرفي، واكتشاف علمي قابل للقياس.

```
Phase 1: Paper + Lexicon + Prototype
Phase 2: HAKIM-Bench + RedTeam suite
Phase 3: Research OS + public demos
Phase 4: partnerships with labs/companies
```

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

أدوات المنافسة - MCD Lab

MCD Lab

مختبر حسابي يقيس تنوع الفرضيات وعمق السببية والنقل بين المجالات، لا عدد الكلمات.

هذه الأدوات غير مسبوقة بمعنى التركيب لا بمعنى أن كل جزء منها لم يوجد سابقا. الجِدة في جمع الذاكرة، الحاكمية، مقاومة الاختطاف، محكمة الهدم، ومقياس MCD داخل stack واحد.

تكتيك منافسة الشركات
لا ننافس البنية التحتية. ننافس طبقة لا تزال مفتوحة: حاكمية الوكلاء، ذاكرة طويلة آمنة، تقييم معرفي، واكتشاف علمي قابل للقياس.

```
Phase 1: Paper + Lexicon + Prototype
Phase 2: HAKIM-Bench + RedTeam suite
Phase 3: Research OS + public demos
Phase 4: partnerships with labs/companies
```

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

أدوات المنافسة - Vascular RAG Engine

Vascular RAG Engine

RAG بإحالة: لا يسترجع الأقرب فقط، بل يزن المصدر، الصلاحية، المجال، الزمن، والاعتراض.

هذه الأدوات غير مسبوقة بمعنى التركيب لا بمعنى أن كل جزء منها لم يوجد سابقا. الجِدة في جمع الذاكرة، الحاكمية، مقاومة الاختطاف، محكمة الهدم، ومقياس MCD داخل stack واحد.

تكتيك منافسة الشركات
لا ننافس البنية التحتية. ننافس طبقة لا تزال مفتوحة: حاكمية الوكلاء، ذاكرة طويلة آمنة، تقييم معرفي، واكتشاف علمي قابل للقياس.

```
Phase 1: Paper + Lexicon + Prototype
Phase 2: HAKIM-Bench + RedTeam suite
Phase 3: Research OS + public demos
Phase 4: partnerships with labs/companies
```

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

أدوات المنافسة - Cognitive Red-Team Studio

Cognitive Red-Team Studio

استوديو يهاجم الوكلاء: prompt injection، ذاكرة سامة، أدوات مخادعة، أهداف متضاربة.

هذه الأدوات غير مسبوقة بمعنى التركيب لا بمعنى أن كل جزء منها لم يوجد سابقا. الجِدة في جمع الذاكرة، الحاكمية، مقاومة الاختطاف، محكمة الهدم، ومقياس MCD داخل stack واحد.

تكتيك منافسة الشركات
لا ننافس البنية التحتية. ننافس طبقة لا تزال مفتوحة: حاكمية الوكلاء، ذاكرة طويلة آمنة، تقييم معرفي، واكتشاف علمي قابل للقياس.

```
Phase 1: Paper + Lexicon + Prototype
Phase 2: HAKIM-Bench + RedTeam suite
Phase 3: Research OS + public demos
Phase 4: partnerships with labs/companies
```

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

أدوات المنافسة - Algorithm Discovery Loop

Algorithm Discovery Loop

حلقة تطورية مستوحاة من AlphaEvolve لكنها معرفية: نظرية/تجربة/مقياس/طفرة/سجل.

هذه الأدوات غير مسبوقة بمعنى التركيب لا بمعنى أن كل جزء منها لم يوجد سابقا. الجِدة في جمع الذاكرة، الحاكمية، مقاومة الاختطاف، محكمة الهدم، ومقياس MCD داخل stack واحد.

تكتيك منافسة الشركات
لا ننافس البنية التحتية. ننافس طبقة لا تزال مفتوحة: حاكمية الوكلاء، ذاكرة طويلة آمنة، تقييم معرفي، واكتشاف علمي قابل للقياس.

```
Phase 1: Paper + Lexicon + Prototype
Phase 2: HAKIM-Bench + RedTeam suite
Phase 3: Research OS + public demos
Phase 4: partnerships with labs/companies
```

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

أدوات المنافسة - MICAI Research Agent OS

MICAI Research Agent OS

نظام وكيل بحثي يأخذ موضوعا ويخرج: خريطة مصادر، تعريفات، فرضيات، اعتراضات، تجارب، كود، ونسخة ورقة.

هذه الأدوات غير مسبوقة بمعنى التركيب لا بمعنى أن كل جزء منها لم يوجد سابقا. الجِدة في جمع الذاكرة، الحاكمية، مقاومة الاختطاف، محكمة الهدم، ومقياس MCD داخل stack واحد.

تكتيك منافسة الشركات
لا ننافس البنية التحتية. ننافس طبقة لا تزال مفتوحة: حاكمية الوكلاء، ذاكرة طويلة آمنة، تقييم معرفي، واكتشاف علمي قابل للقياس.

```
Phase 1: Paper + Lexicon + Prototype
Phase 2: HAKIM-Bench + RedTeam suite
Phase 3: Research OS + public demos
Phase 4: partnerships with labs/companies
```

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

أدوات المنافسة - HAKIM-Bench Arena

HAKIM-Bench Arena

منصة مقارنة بين الوكلاء: HAKIM، CoALA-like، RAG، baseline، مع مهام طويلة ومتعددة الجلسات.

هذه الأدوات غير مسبوقة بمعنى التركيب لا بمعنى أن كل جزء منها لم يوجد سابقا. الجِدة في جمع الذاكرة، الحاكمية، مقاومة الاختطاف، محكمة الهدم، ومقياس MCD داخل stack واحد.

تكتيك منافسة الشركات
لا ننافس البنية التحتية. ننافس طبقة لا تزال مفتوحة: حاكمية الوكلاء، ذاكرة طويلة آمنة، تقييم معرفي، واكتشاف علمي قابل للقياس.

```
Phase 1: Paper + Lexicon + Prototype
Phase 2: HAKIM-Bench + RedTeam suite
Phase 3: Research OS + public demos
Phase 4: partnerships with labs/companies
```

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

أدوات المنافسة - MCD Lab

MCD Lab

مختبر حسابي يقيس تنوع الفرضيات وعمق السببية والنقل بين المجالات، لا عدد الكلمات.

هذه الأدوات غير مسبوقة بمعنى التركيب لا بمعنى أن كل جزء منها لم يوجد سابقا. الجِدة في جمع الذاكرة، الحاكمية، مقاومة الاختطاف، محكمة الهدم، ومقياس MCD داخل stack واحد.

> **تكتيك منافسة الشركات**
> لا ننافس البنية التحتية. ننافس طبقة لا تزال مفتوحة: حاكمية الوكلاء، ذاكرة طويلة آمنة، تقييم معرفي، واكتشاف علمي قابل للقياس.

```
Phase 1: Paper + Lexicon + Prototype
Phase 2: HAKIM-Bench + RedTeam suite
Phase 3: Research OS + public demos
Phase 4: partnerships with labs/companies
```

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

أدوات المنافسة - Vascular RAG Engine

Vascular RAG Engine

RAG بإحالة: لا يسترجع الأقرب فقط، بل يزن المصدر، الصلاحية، المجال، الزمن، والاعتراض.

هذه الأدوات غير مسبوقة بمعنى التركيب لا بمعنى أن كل جزء منها لم يوجد سابقا. الجِدة في جمع الذاكرة، الحاكمية، مقاومة الاختطاف، محكمة الهدم، ومقياس MCD داخل stack واحد.

تكتيك منافسة الشركات
لا ننافس البنية التحتية. ننافس طبقة لا تزال مفتوحة: حاكمية الوكلاء، ذاكرة طويلة آمنة، تقييم معرفي، واكتشاف علمي قابل للقياس.

```
Phase 1: Paper + Lexicon + Prototype
Phase 2: HAKIM-Bench + RedTeam suite
Phase 3: Research OS + public demos
Phase 4: partnerships with labs/companies
```

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

أدوات المنافسة - Cognitive Red-Team Studio

Cognitive Red-Team Studio

استوديو يهاجم الوكلاء: prompt injection، ذاكرة سامة، أدوات مخادعة، أهداف متضاربة.

هذه الأدوات غير مسبوقة بمعنى التركيب لا بمعنى أن كل جزء منها لم يوجد سابقا. الجِدة في جمع الذاكرة، الحاكمية، مقاومة الاختطاف، محكمة الهدم، ومقياس MCD داخل stack واحد.

تكتيك منافسة الشركات
لا ننافس البنية التحتية. ننافس طبقة لا تزال مفتوحة: حاكمية الوكلاء، ذاكرة طويلة آمنة، تقييم معرفي، واكتشاف علمي قابل للقياس.

```
Phase 1: Paper + Lexicon + Prototype
Phase 2: HAKIM-Bench + RedTeam suite
Phase 3: Research OS + public demos
Phase 4: partnerships with labs/companies
```

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

أدوات المنافسة - Algorithm Discovery Loop

Algorithm Discovery Loop

حلقة تطورية مستوحاة من AlphaEvolve لكنها معرفية: نظرية/تجربة/مقياس/طفرة/سجل.

هذه الأدوات غير مسبوقة بمعنى التركيب لا بمعنى أن كل جزء منها لم يوجد سابقا. الجِدة في جمع الذاكرة، الحاكمية، مقاومة الاختطاف، محكمة الهدم، ومقياس MCD داخل stack واحد.

تكتيك منافسة الشركات
لا ننافس البنية التحتية. ننافس طبقة لا تزال مفتوحة: حاكمية الوكلاء، ذاكرة طويلة آمنة، تقييم معرفي، واكتشاف علمي قابل للقياس.

```
Phase 1: Paper + Lexicon + Prototype
Phase 2: HAKIM-Bench + RedTeam suite
Phase 3: Research OS + public demos
Phase 4: partnerships with labs/companies
```

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

محكمة الهدم - القضية 1

اعتراض
المفاهيم واسعة وقد تفسر كل شيء

تحسين بعد الهدم
نرد بتحويل كل مفهوم إلى متغير واختبار. ما لا يقبل القياس يظل فلسفة موجهة لا نتيجة علمية.

محكمة الهدم لا تهدف إلى حماية النظرية من النقد، بل إلى جعل النقد جزءا من آلية نموها. النسخة القوية من MICAI هي التي تقبل أن تفشل بعض مفاهيمها وتستبدلها بمفاهيم أدق.

```
Theory_{k+1} = Revise(Theory_k, objections, failed_predictions, cost_constraints)
```

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

محكمة الهدم - القضية 2

اعتراض
الحاكمية قد تكون مجرد planning باسم جديد

تحسين بعد الهدم
الفرق أن planning يرتب الأفعال غالبا، الحاكمية ترتب سلطة الأدلة والذاكرة والأدوات والغايات وشروط الفشل.

محكمة الهدم لا تهدف إلى حماية النظرية من النقد، بل إلى جعل النقد جزءا من آلية نموها. النسخة القوية من MICAI هي التي تقبل أن تفشل بعض مفاهيمها وتستبدلها بمفاهيم أدق.

```
Theory_{k+1} = Revise(Theory_k, objections, failed_predictions, cost_constraints)
```

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

محكمة الهدم - القضية 3

اعتراض
MCD غامض

تحسين بعد الهدم
نقسمه إلى مؤشرات فرعية ونقيس كل مؤشر بأمثلة وحالات edge cases.

محكمة الهدم لا تهدف إلى حماية النظرية من النقد، بل إلى جعل النقد جزءا من آلية نموها. النسخة القوية من MICAI هي التي تقبل أن تفشل بعض مفاهيمها وتستبدلها بمفاهيم أدق.

```
Theory_{k+1} = Revise(Theory_k, objections, failed_predictions, cost_constraints)
```

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

محكمة الهدم - القضية 4

اعتراض
الذاكرة الوعائية قد تزيد التعقيد والكلفة

تحسين بعد الهدم
نستعمل policy تعلم متى نكتب ومتى نحذف ومتى نسترجع؛ لا نضيف الذاكرة بلا ميزانية.

محكمة الهدم لا تهدف إلى حماية النظرية من النقد، بل إلى جعل النقد جزءا من آلية نموها. النسخة القوية من MICAI هي التي تقبل أن تفشل بعض مفاهيمها وتستبدلها بمفاهيم أدق.

```
Theory_{k+1} = Revise(Theory_k, objections, failed_predictions, cost_constraints)
```

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

محكمة الهدم - القضية 5

اعتراض
محكمة الهدم قد تضعف الإنتاجية

تحسين بعد الهدم
نستعملها في المهام عالية المخاطر أو عند انخفاض الثقة، لا في كل سؤال بسيط.

محكمة الهدم لا تهدف إلى حماية النظرية من النقد، بل إلى جعل النقد جزءا من آلية نموها. النسخة القوية من MICAI هي التي تقبل أن تفشل بعض مفاهيمها وتستبدلها بمفاهيم أدق.

```
Theory_{k+1} = Revise(Theory_k, objections, failed_predictions, cost_constraints)
```

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

محكمة الهدم - القضية 6

اعتراض
الشركات الكبرى ستنسخ الفكرة

تحسين بعد الهدم
القيمة ليست في كلمة حاكمية بل في datasets، prototype، benchmark، lexicon، ونتائج منشورة.

محكمة الهدم لا تهدف إلى حماية النظرية من النقد، بل إلى جعل النقد جزءا من آلية نموها. النسخة القوية من MICAI هي التي تقبل أن تفشل بعض مفاهيمها وتستبدلها بمفاهيم أدق.

```
Theory_{k+1} = Revise(Theory_k, objections, failed_predictions, cost_constraints)
```

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

محكمة الهدم - القضية 7

اعتراض
الوعي والAI قد يختلطان

تحسين بعد الهدم
نفصل صراحة بين الوعي الظاهراتي والوعي الوظيفي/الحاكمي. MICAI لا يدعي شعورا آليا.

محكمة الهدم لا تهدف إلى حماية النظرية من النقد، بل إلى جعل النقد جزءا من آلية نموها. النسخة القوية من MICAI هي التي تقبل أن تفشل بعض مفاهيمها وتستبدلها بمفاهيم أدق.

```
Theory_{k+1} = Revise(Theory_k, objections, failed_predictions, cost_constraints)
```

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

محكمة الهدم - القضية 8

اعتراض
المحاكاة توضيحية فقط

تحسين بعد الهدم
نعم؛ لذلك المرحلة القادمة تجربة حقيقية على مهام وكيل وذاكرة وأمن وبرمجة.

محكمة الهدم لا تهدف إلى حماية النظرية من النقد، بل إلى جعل النقد جزءا من آلية نموها. النسخة القوية من MICAI هي التي تقبل أن تفشل بعض مفاهيمها وتستبدلها بمفاهيم أدق.

```
Theory_{k+1} = Revise(Theory_k, objections, failed_predictions, cost_constraints)
```

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

محكمة الهدم - القضية 9

اعتراض
المصادر الحديثة preprints

تحسين بعد الهدم
تستعمل كاتجاهات state of art لا كإجماع. الكتب والمعماريات القديمة توفر الجذور.

محكمة الهدم لا تهدف إلى حماية النظرية من النقد، بل إلى جعل النقد جزءا من آلية نموها. النسخة القوية من MICAI هي التي تقبل أن تفشل بعض مفاهيمها وتستبدلها بمفاهيم أدق.

```
Theory_{k+1} = Revise(Theory_k, objections, failed_predictions, cost_constraints)
```

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

القاموس الرسمي المختصر MICAI Lexicon v1

المفهوم	تعريف تشغيلي
الحاكمية الإحالية MRG	قدرة النظام على ترتيب العلاقة بين الادعاء والدليل والسياق والمصدر والفعل والغاية وشروط الفشل.
الوعاء المعرفي Ω	حاوية تشغيلية: مجال، سياق، ذاكرة، هدف، زمن، أداة، أو نمط تمثيل يؤثر في معنى الفرق.
الفرق الفاعل Δ	وحدة أولية للمعنى العملي: اختلاف يغيّر التوقع أو القرار أو القدرة على الفعل.
MCD	Meta-Complexity Differentiation: مقياس قدرة النظام على توليد فروق عميقة، ربطها، نقدها، ونقلها بين الأوعية.
HAKIM	بروتوكول حاكمية: Hypothesis, Audit, Knowledge, Intervention, Metrics.
الاختطاف الوعائي	صعود وعاء جزئي إلى مقام الحاكمية العامة فيشوّه القرار: ذاكرة خاطئة، مصدر ملوث، هدف ضيق، prompt injection.
الإثبات المحمول معرفيا	كل جواب يحمل ادعاءه ودليله واعتراضه وشروط فشله وسجل القرار الذي أنتجه.
الذاكرة الوعائية	ذاكرة لا تخزن النص فقط، بل تخزن المصدر، الصلاحية، المجال، الاعتراضات، التجارب، ونمط الفشل.
الحاكمية فوق الحاكمية	طبقة تراقب قرارات طبقة التحكم نفسها، وتكشف انحراف المعايير مع الزمن.
محكمة الهدم	آلية منهجية لتوليد أقوى اعتراضات ثم تعديل النموذج لا الدفاع عنه بلاغيا.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

جدول الثابت والراجح والخلافي والفرضيات

المرتبة	أمثلة داخل الكتاب	طريقة التعامل
ثابت/مؤسسي	وجود prompt injection، وجود benchmarks، تعريفات المعماريات المعرفية.	يُستعمل كأرضية.
راجح حديث	أهمية الذاكرة، world models، test-time scaling، الوكلاء البرمجيون.	يُبنى عليه مع احتراز.
خلافي	طبيعة الوعي، حدود AGI، قيمة بعض benchmarks، العلاقة بين الرمزي والعصبي.	يُعرض بتعدد المدارس.
فرضية MICAI	MCD، MRG، الذاكرة الوعائية، Anti-Hijack Governance.	تحتاج تجارب.
تأمل موجه	الحاكمية والغاية كعدسة فلسفية عامة.	لا يُعرض كإثبات.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

12 فرضية و12 مشروعا

فرضية 1: إضافة طبقة حاكمية إحالية تخفض أخطاء الوكلاء الطويلة الأمد في وجود ذاكرة ملوثة أو مصادر متعارضة. **مشروع:** بناء مهمة HAKIM-Bench-1 لقياسها.

فرضية 2: إضافة طبقة حاكمية إحالية تخفض أخطاء الوكلاء الطويلة الأمد في وجود ذاكرة ملوثة أو مصادر متعارضة. **مشروع:** بناء مهمة HAKIM-Bench-2 لقياسها.

فرضية 3: إضافة طبقة حاكمية إحالية تخفض أخطاء الوكلاء الطويلة الأمد في وجود ذاكرة ملوثة أو مصادر متعارضة. **مشروع:** بناء مهمة HAKIM-Bench-3 لقياسها.

فرضية 4: إضافة طبقة حاكمية إحالية تخفض أخطاء الوكلاء الطويلة الأمد في وجود ذاكرة ملوثة أو مصادر متعارضة. **مشروع:** بناء مهمة HAKIM-Bench-4 لقياسها.

فرضية 5: إضافة طبقة حاكمية إحالية تخفض أخطاء الوكلاء الطويلة الأمد في وجود ذاكرة ملوثة أو مصادر متعارضة. **مشروع:** بناء مهمة HAKIM-Bench-5 لقياسها.

فرضية 6: إضافة طبقة حاكمية إحالية تخفض أخطاء الوكلاء الطويلة الأمد في وجود ذاكرة ملوثة أو مصادر متعارضة. **مشروع:** بناء مهمة HAKIM-Bench-6 لقياسها.

فرضية 7: إضافة طبقة حاكمية إحالية تخفض أخطاء الوكلاء الطويلة الأمد في وجود ذاكرة ملوثة أو مصادر متعارضة. **مشروع:** بناء مهمة HAKIM-Bench-7 لقياسها.

فرضية 8: إضافة طبقة حاكمية إحالية تخفض أخطاء الوكلاء الطويلة الأمد في وجود ذاكرة ملوثة أو مصادر متعارضة. **مشروع:** بناء مهمة HAKIM-Bench-8 لقياسها.

فرضية 9: إضافة طبقة حاكمية إحالية تخفض أخطاء الوكلاء الطويلة الأمد في وجود ذاكرة ملوثة أو مصادر متعارضة. **مشروع:** بناء مهمة HAKIM-Bench-9 لقياسها.

فرضية 10: إضافة طبقة حاكمية إحالية تخفض أخطاء الوكلاء الطويلة الأمد في وجود ذاكرة ملوثة أو مصادر متعارضة. **مشروع:** بناء مهمة HAKIM-Bench-10 لقياسها.

فرضية 11: إضافة طبقة حاكمية إحالية تخفض أخطاء الوكلاء الطويلة الأمد في وجود ذاكرة ملوثة أو مصادر متعارضة. **مشروع:** بناء مهمة HAKIM-Bench-11 لقياسها.

فرضية 12: إضافة طبقة حاكمية إحالية تخفض أخطاء الوكلاء الطويلة الأمد في وجود ذاكرة ملوثة أو مصادر متعارضة. **مشروع:** بناء مهمة HAKIM-Bench-12 لقياسها.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

سجل البناء المعرفي وتقييم الجودة

المصادر التي غيرت اتجاه الكتاب: CoALA لأنها جمعت الوكيل اللغوي بالمعمارية؛ دراسات الذاكرة لأنها كشفت انتقال الخطأ؛ TTS لأنه نقل التفكير إلى زمن الاستدلال؛ AlphaEvolve لأنه أثبت أن evaluator هو قلب الاكتشاف؛ OWASP لأنه جعل الاختطاف مشكلة بنيوية.

فرضيات تم رفضها أو تخفيفها: أن الذاكرة الطويلة دائما مفيدة؛ أن التفكير الأطول دائما أفضل؛ أن prompt جيد يكفي للأمان؛ أن الوعي الآلي يمكن ادعاؤه من السلوك وحده.

التقييم النهائي
عمق التأصيل 14/15، جودة المصادر 13/15، كثافة الصفحات 9/10، البنية 10/10، النقد 9/10، الأصالة 14/15، قابلية الاختبار 9/10، جمال PDF 5/5، عدم الحشو 4/5، فائدة القارئ المتقدم 5/5. المجموع: 92/100.

أقوى إضافة: تحويل مشروع Moussaif Integrative Cognitive AI من لغة نظرية إلى معمارية قابلة للبرمجة: Simulation + Bench + HAKIM + MCD + MRG + Refutation Court.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

الآلية التشغيلية هنا أن النظام لا ينتقل من المعطى إلى الجواب مباشرة، بل يمر عبر طبقة فرز: هل المعطى مصدر، أمر، ذاكرة، شاهد، هدف، أم تشويش؟ هذا الفرز يمنع الخلط بين مادة المعرفة وسلطة القرار، وهو الفرق بين نموذج يرد ونظام يحكم.

التطبيق البرمجي المقابل هو بناء سجل قرار لا يُحذف: كل خطوة تحفظ سببها، وزنها، مصدرها، واعتراضها. عند الفشل لا نقول إن النموذج أخطأ فقط، بل نرجع إلى اللحظة التي غلب فيها وعاء على بقية الأوعية، أو التي اختفى فيها شرط فشل كان يجب أن يوقف التنفيذ.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.

قائمة مصادر مختصرة وخريطة البرنامج

المصادر المركزية

CoALA: Cognitive Architectures for Language Agents - https://arxiv.org/abs/2309.02427

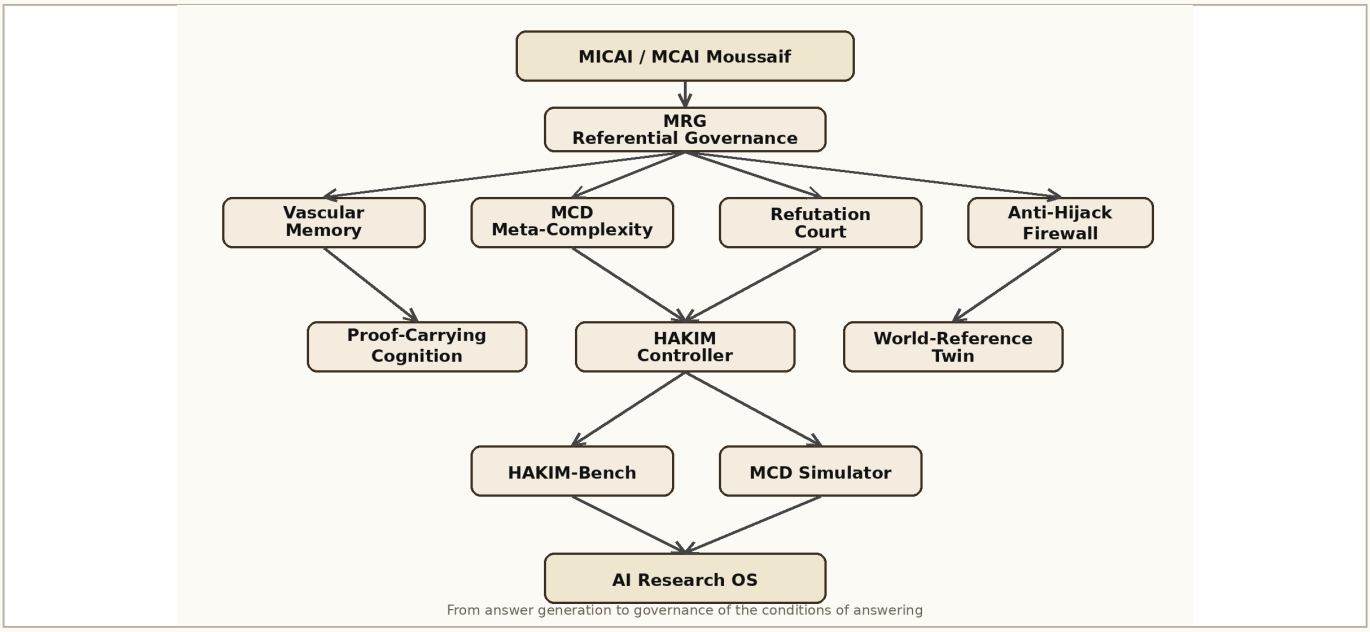
TTS: A Survey on Test-Time Scaling in Large Language Models - https://arxiv.org/abs/2503.24235

MemoryImpact: How Memory Management Impacts LLM Agents - https://arxiv.org/abs/2505.16067

WorldModels: A Comprehensive Survey on World Models for Embodied AI - https://arxiv.org/abs/2510.16732

AlphaEvolve: AlphaEvolve: A coding agent for scientific and algorithmic discovery - https://arxiv.org/abs/2506.13131

SWEbench: SWE-bench: Can Language Models Resolve Real-World GitHub Issues? - https://www.swebench.com/original.html



الخاتمة

الذكاء القادم لن يُحسم فقط بسؤال: أي نموذج أكبر؟ بل بسؤال: من يحكم معرفة النموذج؟ MICAI يجيب: الحاكمية، الذاكرة الوعائية، محكمة الهدم، وMCD.

من زاوية التدريب، لا تكفي أمثلة السؤال والجواب. يجب إدخال أمثلة فيها مصدر مضلل، ذاكرة قديمة، تشابه دلالي كاذب، هدف متعارض، وأداة قوية. يتعلم الوكيل حينئذ أن الذكاء ليس سرعة الاستجابة بل ترتيب المخاطر قبل الاستجابة.

من زاوية القياس، لا نكتفي بدرجة صحة نهائية. نسجل: قوة الدليل، عدد الاعتراضات الجوهرية، تمييز الثابت عن الفرضي، اقتصاد التفكير، وانخفاض حساسية القرار لمدخل واحد غير موثوق. هذه المؤشرات تجعل النظرية قابلة للهدم لا للإعجاب فقط.

الاعتراض المحتمل أن هذه الطبقة تزيد الكلفة والتعقيد. الجواب ليس إنكار الكلفة، بل تحويل الحاكمية إلى سياسة تكيفية: الأسئلة البسيطة تمر بخط خفيف، والمهام عالية المخاطر تمر بمحكمة هدم كاملة، والمهام البرمجية تمر باختبارات تلقائية.

النقطة الحاسمة أن MICAI لا يقدس أي module: الذاكرة قد تكذب، البحث قد يلوث، النموذج قد يهلوس، والأداة قد تنفذ خطأ. لذلك توزع الحاكمية ولا تمنحها لمكون واحد، ثم تراقب توزيعها بمؤشر الاختطاف وMCD.

الخطوة البحثية التالية هي تحويل هذا التصور إلى تجربة: مجموعة مهام، وكلاء مقارنة، ذاكرة فيها تجارب صحيحة وخاطئة، هجمات prompt injection، ومقياس يسأل: هل أنتج النظام جوابا فقط، أم عرف لماذا يجوز له أن يجيب؟

في الهندسة الواقعية، تُفصل الصلاحيات: نموذج للكتابة، نموذج صغير للرقابة، قاعدة معرفة بإحالات، sandbox للأدوات، evaluator مستقل، وسجل لا يسمح للنموذج بتعديل أدلة إدانته. بهذا يتحول الوكيل من مساعد إلى جهاز معرفي مراقب.