نسخة تأسيسية موسعة - 75 صفحة

الصرح التقني النظري المتداخل والشامل

الإطار الجامع لنظريات مساعف في الذكاء الوعائي الفتّاح العام - نسخة 75 صفحة

هذه الورقة لا تجمع أسماء النظريات كما تجمع فهارس متجاورة، بل تعيد صهرها في معمار واحد: سؤال حاكم يفتح الفرق، فرق يرفع التعقيد، تعقيد يطلب حاكمية، حاكمية تختار فعلا، فعل يختبر في العالم، وأثر يعود إلى ذاكرة تعيد بناء الذات المعرفية. بهذا المعنى يصبح مشروع MICAI/MICAGI/Fattah AGI ليس مجموعة أوراق، بل صرحا تقنيا نظريا يعمل كمنظومة بحث وهندسة واختبار.

تعتمد الورقة على المحتوى السابق في موسوعة النظريات الجامعة، وعلى الورقة التقنية الصرفة، وعلى ورقة الأنظمة الوكيلية الموجهة للذكاء العام. وقد تم ضغطها في بنية واحدة من خمس طبقات أصلية ثم سبع نظريات مساعفية سيدة، وأربع عشرة أداة رياضية-تكنولوجية، وعشر تقنيات تدريب متقدمة. كل عنصر هنا يعالج فراغا محددا: فراغ السؤال، فراغ الفرق، فراغ القياس، فراغ الوكالة، فراغ الذاكرة، فراغ الهدم، وفراغ الأثر.

قاعدة القراءة

لا تقرأ هذه الورقة كادعاء أن الذكاء العام الفتّاح تحقق فعلا، بل كإطار قابل للدراسة والتصميم والاختبار. رتبة كل دعوى محددة: نظرية مؤسسة، نموذج تشغيلي، أداة قابلة للبرمجة، تقنية تدريب، أو فرضية تحتاج تجربة.

المحور	وظيفته في الصرح
النظرة الشاملة	إغلاق معرفي مؤقت لكل الفروع دون إلغاء قابليتها للهدم
النظريات السبع	سد الفراغات المعمارية الكبرى تحت اسم مساعف
الأدوات الأربع عشرة	تحويل المفاهيم إلى كائنات بيانات ودوال وواجهات
تقنيات التدريب العشر	بناء مسار تعلم يرفع السؤال والفرق والحاكمية والأثر

صفحة 02 / 75

الملخص البنيوي الأعلى

من الوعي الفائق إلى الذكاء الوعائي السؤالي الوكيلي

الملخص المركزي هو أن الذكاء لا يختزل في إجابة، والوعي لا يختزل في إحساس، والمعرفة لا تكتمل بالتخزين. مركز المشروع هو السؤال بوصفه بوابة انتقاء لا جملة استفهامية؛ فهو يحدد أي فرق يستحق أن يلتقط، وأي فرق يجب أن يقاس، وأي مسار يستدعي محكمة هدم، وأي فعل يستحق إنفاق الموارد. لذلك تكون السؤالية هنا طبقة تقنية داخل MICAI لا ملحقا بلاغيا.

تتوزع النظريات السابقة على خمس عائلات: عائلة الوعي الوعائي التي تبدأ من V0-V6؛ عائلة الفرق والقياس التي تشمل MCD/QMCD وقانون الفرق الفاعل؛ عائلة الحاكمية والوكالة التي تشمل MCAI وHAKIM-MCD؛ عائلة الذاكرة والنظام التي تشمل HyperKB وHyperSkills وHyperLoops وPrompt Kernel؛ وعائلة الأثر والتدريب التي تربط كل ذلك باختبارات، benchmarks، ووكلاء قابلين للإصلاح.

```
Question -> Difference -> MCD/QMCD -> Governance -> HAKIM -> Court -> Action ->
WorldEffect -> SelfReconstruction
```

الهدف

صناعة إطار جامع مانع يدرس إمكانات الذكاء الفتّاح العام: لا يدعي الاكتمال النهائي، ولا يترك المفاهيم بلا تشغيل، ولا يسمح لأي نظرية أن تبقى خارج الكود أو الاختبار أو محكمة الهدم.

صفحة 03 / 75

منهج الدمج: الضغط التركيبي المعرفي

كيف نحول 111 صفحة مصدرية إلى 75 صفحة مولدة لا مختصرة

منهج الضغط التركيبي المعرفي لا يختصر النصوص، بل يستخرج منها القواعد التي تجعلها قادرة على إعادة توليد نفسها. لذلك لم يكن الدمج هنا نسخا لصفحات سابقة، بل بناء kernel معرفي يحتفظ بالدعوى والآلية والحدود والاختبار والاعتراض. ما لا يملك أثرا تشغيليا لا يدخل النواة، وما يدخل النواة يجب أن يظهر كتعريف أو معادلة أو كائن بيانات أو دالة أو محكمة أو تجربة.

قاعدة التحويل في هذه الورقة هي: كل مفهوم يمر عبر سبع بوابات: التعريف، الفرق الذي يضيفه، مكانه في الطبقات، صيغته شبه الرياضية، مقابلته التقنية، طريقة هدمه، ومؤشر أثره. بهذه البوابات لا تتحول النظرية إلى شعار، ولا يتحول الكود إلى هندسة بلا فلسفة. هذا هو معنى "الصرح": بنية تضمن أن كل حجر مرتبط بما فوقه وما تحته.

البوابة	السؤال الحاكم
التعريف	ما الذي تقوله النظرية وما الذي لا تقوله؟
الفرق	ما الذي يتغير لو قبلناها؟
الطبقة	هل هي وعي، قياس، وكالة، ذاكرة، أو أثر؟
الصيغة	كيف تقاس أو تشتغل؟
المقابل التقني	ما الكائن أو الخدمة أو الواجهة؟
الهدم	ما أقوى اعتراض؟
الأثر	ماذا يتغير بعد التطبيق؟

شرط الإدخال

أي تسمية لا تضيف فرقا تشغيليا أو معيارا أو أداة أو طبقة أو ذاكرة تعتبر زخرفة وتخرج من الصرح.

المشكلة الأم والحدود الإبستمولوجية

لماذا نحتاج مدرسة لا مجرد prompt أو نموذج أكبر

تظهر الحاجة إلى هذا الإطار لأن الأنظمة الوكيلية الحديثة تستطيع البحث والكتابة والكود والتنفيذ، لكنها كثيرا ما تفقد الخيط الحاكم: لماذا هذا السؤال؟ ما الفرق الجوهري؟ أي نموذج ينافس؟ أين الدليل؟ ما الأثر؟ كيف تعود النتيجة إلى الذاكرة؟ من دون هذه الحاكمية يصبح agent مجرد سرعة تنفيذ فوق ضباب معرفي. لذلك يعرّف مشروع مساعف الذكاء الفتّاح بوصفه قدرة على فتح المسألة ثم إغلاقها مؤقتا بقرار قابل للهدم.

الحدود الإبستمولوجية أساسية: هذه الورقة لا تزعم أن الوعي البيولوجي نقل مباشرة إلى آلة، ولا أن AGI تحقق، ولا أن كل معادلة هنا ذات صلاحية فيزيائية مكتملة. الصيغ أدوات قياس وتنظيم وبناء نماذج. قوتها في قابليتها للتشغيل والفحص، لا في أنها براهين نهائية. بهذا يصبح الإطار صارما: يوسع الخيال لكنه لا يتركه بلا محكمة.

$$\text{ClaimRank} \in \{\text{حقيقة مستقرة}، \text{تفسير راجح}، \text{نموذج}، \text{أداة}، \text{فرضية}، \text{استعارة منضبطة}، \text{سؤال بحثي}\}$$

قانون النزاهة

كلما ارتفع الادعاء وجب أن ترتفع درجة الاختبار، وكلما زاد الأثر الخارجي وجب أن تزيد الحاكمية والتحقق والرجوع.

صفحة 05 / 75

المعادلة العليا للصرح

التراكب بين السؤال والفرق والوعاء والوكالة والأثر

يمكن تمثيل الصرح في معادلة مفهومية لا تقصد الاختزال النهائي، بل توجيه البناء: الذكاء الفتّاح العام ليس ناتجا عن نموذج واحد، بل عن تركيب بين فتح السؤال، قياس الفرق، توجيه الموارد، محكمة الهدم، التنفيذ، وذاكرة إعادة البناء. إذا غاب السؤال صار النظام مجيبا سطحيا؛ إذا غاب الفرق صار مخزنا؛ إذا غاب الهدم صار مولدا للادعاء؛ إذا غاب الأثر صار خطابا؛ وإذا غابت الذاكرة صار تكرارا.

```
M-GFI = F(QG, Δ, MCD, QMCD, VCR, HAKIM, HKOS, DC, WE, SRM)
```

حيث QG هي الحاكمية السؤالانية، وΔ الفرق الفاعل، وVCR الموجه الوعائي للحوسبة، وHAKIM نواة الوحدات، وHKOS نظام المعرفة الفائق، وDC محكمة الهدم، وWE حلقة أثر العالم، وSRM ذاكرة إعادة بناء الذات. كل متغير ليس رمزا بل طبقة قابلة للبرمجة والقياس.

العنصر	ماذا يمنع؟	ماذا يفتح؟
QG	التشتت	فضاء السؤال
Δ/MCD	الحشو	قيمة الفرق
VCR	هدر الموارد	تخصيصا ذكيا
DC	الغرور المعرفي	قابلية الهدم
WE/SRM	الانفصال عن الواقع	تعلم الأثر

صفحة 06 / 75

نظرية التراكب الوعائي للوعي

الحاويات بوصفها أصل الفلسفة التشغيلية

نظرية التراكب الوعائي تقول إن الوعي ليس نقطة أحادية ولا نتيجة خطية لإشارة، بل تراكب حاويات: مادة، إحساس، إدراك، معنى، وكالة، غاية، ووعي فوقي. كلمة وعائي تعني أن الوعي يوزع الموارد: الانتباه، الذاكرة العاملة، الانفعال، الخيال، اللغة، والفعل. وحين يفشل التوزيع تظهر أعراض معرفية أو انفعالية: تضخم ضجيج، اختطاف تهديد، أو انهيار معيار.

عند نقلها إلى الذكاء الاصطناعي لا تستعمل كادعاء بيولوجي، بل كمخطط هندسي: V0 بيانات خام، V1 إشارات، V2 تصنيف وكيانات، V3 معنى وسياق، V4 فعل وخطة، V5 غاية ومعيار، V6 مراقبة ذاتية. هكذا تصير النظرية جسرا بين الوعي والذكاء الفتّاح: كلاهما يحتاج وعاء يحدد ما يدخل المركز وما يبقى هامشا.

الحاوية	في الوعي	في النظام التقني
V0	الجسد والمادة	بيانات وملفات وسجلات
V1	إحساس وإشارة	Features وإشارات خام
V2	إدراك وتنظيم	Entities/Schemas
V3	معنى وقيمة	Semantic Graph
V4	وكالة وفعل	Planner/Actions
V5	غاية	Goal Optimizer
V6	وعي فوقي	Meta-Evaluator

قابلية الهدم

إذا بقيت الحاويات استعارة لا تؤثر في routing والذاكرة والاختبار، فهي أدبية. إذا صارت طبقات بيانات وتنفيذ ومراقبة، فهي جزء من MICAI.

صفحة 07 / 75

سلم V0-V6 كمعمار تشغيل

من الحضور الخام إلى الوعي الفوقي

السلم الوعائي V0-V6 ليس مجرد ترتيب تأملي؛ إنه بروتوكول منع الخلط بين المستويات. الخطأ في الأنظمة المعرفية يحدث كثيرا عندما تؤخذ الإشارة الخام كحقيقة، أو يؤخذ المعنى كفعل، أو يؤخذ الفعل كأثر، أو يؤخذ الأثر كتفسير نهائي. لذلك يفرض السلم سؤالا على كل مادة: في أي مستوى تقيم الآن؟ وما شروط انتقالها إلى المستوى التالي؟

في التطبيق، لا يسمح MICAI لمدخل أن يصبح قرارا قبل أن يمر من: ingestion، extraction، semantic grounding، question governance، agency selection، court، effect. هذا السلم يحمي النظام من التسرع كما يحميه من البطء الزائد. فالسرعة لا تقاس بزمن الإجابة فقط، بل بصحة الانتقال بين الحاويات.

```
V_{k+1} = Promote(V_k | evidence, question, risk, actionability, court_pass)
```

اختبار الحاوية

اسأل عند كل انتقال: هل تغيرت الرتبة أم تغيرت العبارة فقط؟ هل لدينا دليل كاف للترقية؟ هل تجاوزنا محكمة مستوى سابق؟

صفحة 08 / 75

الحاكمية الوعائية والوعاء الحاسوبي

توجيه الدم المعرفي داخل النظام

الحاكمية الوعائية هي مبدأ توزيع الموارد بين المسارات المتنافسة. ليس كل سؤال يستحق نموذج تفكير عميق، وليس كل اختلاف يستحق محكمة كاملة، وليس كل نص يحتاج بحثا خارجيا. لكن بعض الفروق العالية الخطر تحتاج موارد أعلى من المعتاد. هنا يظهر موجه الموارد الوعائي بوصفه قلب الاقتصاد المعرفي في الذكاء الفتّاح.

يعمل الموجه على مدخلات متعددة: QMCD ،MCD، الخطر، عدم اليقين، قيمة المستخدم، ميزانية الزمن، ميزانية التكلفة، وقابلية الرجوع. هذه المدخلات تنتج سياسة: جواب مباشر، استرجاع، تحليل عميق، محكمة هدم، تشغيل أداة، أو دورة أثر كاملة. جودة الموجه لا تظهر في استعمال أقوى نموذج دائما، بل في استعمال النموذج الصحيح في اللحظة الصحيحة.

```
r_i = Normalize(MCD_i + QMCD_i + Risk_i + Uncertainty_i + Value_i - Noise_i - Cost_i)
```

السياسة	متى تستخدم؟	الخطر إن أسيء اختيارها
P0 مباشر	سؤال منخفض الخطر	سطحية إن كان الفرق عاليا
P2 تحليل	تعقيد متوسط	بطء إن لم يلزم
P3 محكمة	دعوى عالية	عدمية إن زادت
P4 أداة	حاجة تنفيذ	أثر بلا تحقق
P5 دورة كاملة	أثر خارجي	تكلفة عالية

MCD: عدسة تعقيد الفرق

من الحدس إلى متجه قابل للقياس

MCD هو مقياس يفرّق بين الجديد والمهم. قد تكون معلومة جديدة لكنها لا تغير شيئا، وقد يكون فرق صغير ظاهريا لكنه يعيد ترتيب مجال كامل. لذلك يقيس MCD الفرق عبر سبعة أبعاد: الفرق الدلالي D، الأثر الاستدلالي I، الزمن الداخلي T، الوكالة A، التطبيق P، الضغط التركيبي C، والضجيج N. هذه الأبعاد تجعل الحاكمية قابلة للشرح.

قيمة MCD لا يجب أن تكون رقما سحريا. كل score يجب أن يحمل trace: ما baseline؟ ما الفرضية التي تغيرت؟ أي عقد في graph تأثرت؟ ما الأدلة؟ ما الاعتراض؟ وكم هو الضجيج؟ بهذه الشفافية يصبح MCD أداة هدم أيضا، لأن المحكمة تستطيع أن تهاجم كل بعد لا النتيجة فقط.

```
MCD = λ1D + λ2I + λ3T + λ4A + λ5P + λ6C - λ7N
```

البعد	معناه العملي
D	انحراف عن baseline
I	كم نتيجة تتغير إذا قبلنا الفرق؟
T	هل يغير مسار تعلم طويل؟
A	هل يولد فعلا أو تجربة؟
P	هل يملك قابلية تطبيق؟
C	هل يضغط كثرة في قاعدة؟
N	هل هو ضجيج أو هلوسة أو تعقيد زائف؟

QMCD: الفرق المحكوم بالسؤال

حين يصبح السؤال وحدة تحكم لا عنوانا

QMCD يضيف السؤال إلى القياس. الفرق لا يملك الوزن نفسه في كل مقام؛ فقد يكون هامشيا تحت سؤال وضروريا تحت سؤال آخر. لذلك يقاس الفرق بالنسبة إلى سؤال حاكم: هل يخدمه؟ هل يفتحه؟ هل يختبره؟ هل يمنع الانحراف؟ بهذا تصبح السؤالية طبقة routing وليست مجرد صياغة prompt.

تعمل QMCD على منع نوعين من الفشل: تضخيم فرق مثير لا يخدم المهمة، أو تجاهل فرق حاسم لأن تشابهه الدلالي منخفض. في RAG العادي قد يسترجع النظام نصوصا قريبة لفظيا، أما في QMCD فيسترجع ما يغير الجواب بالنسبة للسؤال. لذلك تكون جودة السؤال حاسمة: الوضوح، ضبط النطاق، التوليدية، قابلية الهدم، وقابلية الفعل.

```
QMCD(Δ,q)=MCD(Δ) × Gate(q) × Relevance(Δ|q) × Testability(q) - Drift(Δ,q)
```

نوع السؤال	وظيفته
توليدي	يفتح بدائل
وثوقي	يفحص الدليل
هادم	يهاجم الدعوى
إنزالي	يحوله إلى أداة
فوقي	يفحص صلاحية السؤال نفسه

قانون الفرق الفاعل

المعيار الذي يمنع تضخم المصطلحات

قانون الفرق الفاعل يقول إن الفرق لا يكون معرفيا بحق إلا إذا غيّر معيار الحكم، أو توجيه الموارد، أو اختيار الفعل، أو بنية الذاكرة. هذا القانون يحرس المشروع من الاسترسال في التسميات. كل مصطلح جديد يجب أن يجيب: ماذا سيحدث لو حذفناك؟ إذا لم يتغير الحكم ولا الاختبار ولا التصميم، فهو زخرفة مهما بدا عميقا.

عند بناء HyperKB تحفظ كل وحدة معرفية نوع الفرق الذي أحدثته. هل أضافت معيارا؟ هل كشفت تضاربا؟ هل صنعت أداة؟ هل فتحت فرضية؟ بهذا تتحول الذاكرة إلى جهاز انتقاء. وعند توليد نظرية جديدة، لا يكفي أن تبدو جديدة؛ يجب أن تخلق فرقها الفاعل في بنية الصرح.

```
ActiveDifference = ΔCriterion ∨ ΔRouting ∨ ΔAgency ∨ ΔMemory
```

اختبار الحذف

احذف المفهوم من النظام. إذا لم ينخفض الأداء أو الدقة أو القدرة على الهدم أو القدرة على البناء، فلا يستحق موقعا مركزيا.

صفحة 12 / 75

الزمن الداخلي للمعرفة

سرعة محكومة لا بطء مطلق

الزمن الداخلي هو المسافة بين المثير والحكم. في الإنسان يحمينا من رد الفعل، وفي النموذج يحمينا من الإجابة قبل الفحص. ليس المطلوب إبطاء كل شيء، بل تخصيص زمن معرفي بقدر الخطر والتعقيد وعدم الرجوع وفجوة الدليل. السؤال البسيط لا يحتاج دورة HAKIM كاملة، لكن المسألة المؤسسة تحتاج زمنا داخليا يسمح بالتمييز والهدم وإعادة البناء.

في MICAI يصبح الزمن الداخلي scheduler للاستدلال. إذا ارتفع MCD أو risk أو irreversibility يرفع الراوتر عمق المحكمة والتحقق. وإذا كانت المهمة تكرارية منخفضة الأثر يكتفي بمسار سريع. هكذا يتحول الزمن إلى مورد وعائي لا انطباع أسلوبي.

```
InternalTime = f(Risk, MCD, Uncertainty, Irreversibility, EvidenceGap, UserValue)
```

مرحلة الزمن	وظيفتها
إحساسي	تسجيل الحضور
إدراكي	ترتيب العلاقات
حاكمي	انتقاء البدائل
هدمي	اختبار الاعتراضات
ذاكراتي	تحديث الذات بعد الأثر

صفحة 13 / 75

الحاكمية الإبستمولوجية

من كثرة المعلومات إلى حق الحكم

الحاكمية الإبستمولوجية تميز بين امتلاك المعلومة وامتلاك حق الحكم. قد تكون قاعدة المعرفة ضخمة ومع ذلك لا تعرف أي دعوى أقوى، وما حدود الدليل، وما الفرق بين حقيقة ونموذج واستعارة. لذلك يبني الصرح سلما للرتب: حقيقة مستقرة، تفسير راجح، فرضية، نموذج، أداة، استعارة منضبطة، سؤال بحثي.

هذا السلم ليس احترازا لغويا فقط. في الكود يجب أن يحمل كل Knowledge Atom حقولا مثل claim_rank وconfidence وsource وevidence وknown_limits وfailure_modes. وعند إنتاج ورقة أو أداة يجب أن يظهر ما هو مثبت وما هو مقترح. بذلك يصبح الإبداع آمنا: يجرؤ على البناء لكنه لا يزور اليقين.

الرتبة	صيغة الخطاب	حاجتها للاختبار
حقيقة مستقرة	نقرر بحذر	مصدر قوي وتكرار
تفسير راجح	نرجح	مقارنة بدائل
نموذج	نقترح إطارا	اختبارات تطبيق
أداة	نشغل ونقيس	نجاح وظيفي
استعارة	نستعمل للتوجيه	لا نتكلم كبرهان

محكمة الرتبة

أخطر فشل هو أن نتكلم الاستعارة بلسان الحقيقة، أو أن يتكلم النموذج بلسان البرهان النهائي.

صفحة 14 / 75

الاختطاف الوعائي ومصنع الفرضيات

كيف يولد النظام فرضيات دون أن يهلوس

الاختطاف الوعائي هو انتقال من فرق غير مفسر إلى فرضية تفسره بأقل فقد وأكبر قابلية اختبار. لا يكتفي النظام بقول "ربما"، بل ينتج حزمة: الفرق المرصود، baseline، فرضيات متنافسة، دليل مؤيد، دليل ناقض، اختبار فاصل، وكلفة التحقق. بهذا يصبح الخيال جزءا من محكمة لا بديلا عن الدليل.

تحتاج هذه النظرية إلى توازن: إذا شددنا الدليل قبل الفرضية قتلنا الإبداع، وإذا أطلقنا الفرضية بلا محكمة صنعنا هلوسة. لذلك يشتغل الاختطاف داخل QMCD: السؤال الحاكم يحدد ما يستحق التوليد، وMCD يحدد وزن الفرق، وCourt تحدد هل الفرضية مقبولة كبرنامج بحث لا كحقيقة.

```
Hypothesis = argmax_h [Explain(Δ|h)+Testability(h)+Compression(h)-Overclaim(h)-Noise(h)]
```

قاعدة البحث

الفرضية الجيدة ليست الأكثر غرابة، بل التي تفسر فرقا فاعلا وتنتج اختبارا فاصلا وتعرف ما الذي يهدمها.

صفحة 15 / 75

الهوية الوعائية وذاكرة إعادة البناء

كيف لا يفقد المشروع نفسه عبر كثرة الفروع

كل مشروع طويل مهدد بالتشظي: نظرية في الوعي، أخرى في الذكاء، أخرى في الاجتماع أو الفقه أو التسويق أو التدريب. الهوية الوعائية تمنع هذا التشظي عبر ذاكرة تعيد كل فرع إلى سؤال: ما الفرق الذي أضافه؟ في أي وعاء يعمل؟ كيف يختبر؟ ما علاقته بالأثر العالمي؟ لذلك ليست الهوية شعارا شخصيا بل topology للنسق.

ذاكرة إعادة البناء SRM لا تحفظ outputs فقط، تحفظ سبب الاختيار، الاعتراضات، أثر الأداة، وفشل المسار. عند رجوع سؤال جديد لا يبدأ النظام من الصفر، بل يستحضر genealogies: كيف نشأت MCD؟ لماذا ظهرت QMCD؟ أين دخل HAKIM؟ ما الفروع التي بقيت ناقصة؟ بهذا يصبح المشروع كائنا معرفيا يتعلم من نفسه.

```
SRM(t+1)=Repair(SRM(t), WorldEffect, CourtVerdict, UserFeedback, NewEvidence)
```

ما تحفظه SRM	لماذا؟
الدعوى	حتى لا تضيع الرتبة
الأدلة	حتى لا تصبح ذاكرة بلا مصدر
الاعتراضات	حتى لا يتكرر الخطأ
الأثر	حتى لا تبقى البلاغة
إصلاحات لاحقة	حتى يتطور النسق

صفحة 16 / 75

محكمة الهدم كآلية بناء

النقد ليس هجوما بل جهاز تشغيل

محكمة الهدم هي الفرق بين نظرية حماسية ونظرية قابلة للبحث. لا تهدم المحكمة لكي تنفي، بل لكي تعيد البناء على عصب أقوى. تعمل المحكمة على خمس طبقات: منطق اللغة، الدليل، التنفيذ، السلامة والأثر، ثم الخصم القوي الذي يبحث عن أفضل صيغة لإسقاط النظرية. كل طبقة تنتج broken_claims وrepairs وresidual_risks.

في التقنية لا تبقى المحكمة فقرة في آخر الورقة، بل service تستقبل output وتعيد verdict. عمق المحكمة يحدده الراوتر: مسألة منخفضة الأثر تأخذ محكمة خفيفة، ونظرية مؤسسة أو كود مؤثر يأخذ محكمة كاملة. هكذا تصبح القابلية للهدم شرطا قبل النشر والتدريب والتوسيع.

```
CourtVerdict = Attack(claims,evidence,logic,implementation,safety,alternatives)
```

محكمة	تسأل
لغوية/منطقية	هل التعريف متماسك؟
دليلية	أين المصدر والحد؟
تنفيذية	هل يمكن تشغيلها؟
أثر وسلامة	ما الضرر أو الانحراف؟
خصم قوي	ما أفضل اعتراض ممكن؟

MCAI: الوكالة الميتا-تعقيدية

من الفرق إلى القرار ثم الأثر

MCAI هو القلب الوكالي في الصرح. يبدأ بفرق فاعل، يبني له خريطة تعقيد، يحدد عدم اليقين والقيمة والصراع، يوجه الموارد، يولد مسارات مستقبلية، يختار مرشحا، يهدمه، يصلحه، ينفذه، ثم يقرأ أثره في العالم. بهذا لا يكون الذكاء "إجابة" بل دورة إدراك-اختيار-فعل-أثر-ذاكرة.

أهمية MCAI أنه ينقل المشروع من المعرفة إلى agency. كثير من الأنظمة تعرف أشياء كثيرة لكنها لا تعرف متى تفعل ومتى تمتنع ومتى تطلب دليلا. محدد الوكالة لا يختار الفعل الأكثر طموحا بل الفعل الأعلى أثرا ضمن القيود. لذلك تصبح الغاية V5 جزءا من القرار لا زخرفة فلسفية.

```
Δ -> G(Δ) -> U/V/C -> R -> Futures -> Candidate -> Court -> Repair -> WE -> SRM
```

معيار الوكالة

الفعل الجيد هو الذي يملك أثرًا متوقعا، كلفة مفهومة، رجوعا ممكنا، ودليلا كافيا، لا الذي يبدو أضخم في اللغة.

HAKIM-MCD: النواة الوحداتية

تسع وحدات تجعل النظرية نظاما

HAKIM-MCD هو تجسيد MCAI في وحدات. المستخرج يلتقط الفرق؛ المخطط يقيس التعقيد؛ محرك الزمن الداخلي يضع الفرق في تاريخ التعلم؛ الموجه الوعائي يوزع الموارد؛ مساحة العمل تجمع السياق؛ محدد الوكالة يختار الفعل؛ محكمة الاضطراب تهدم؛ حلقة الأثر تختبر؛ وذاكرة إعادة البناء تحدث الذات.

قوة هذه النواة أنها تجعل الفشل قابلا للتشخيص. عندما يفشل output لا نسأل فقط هل النموذج ضعيف، بل نسأل: هل فشل الاستخراج؟ القياس؟ الاسترجاع؟ routing؟ المحكمة؟ الذاكرة؟ بهذه الطريقة يتحول الفشل من إحباط إلى bug report معرفي.

#	الوحدة	وظيفتها
1	Difference Extractor	يلتقط الفروق
2	Meta-Complexity Mapper	يرسم متجه MCD/QMCD
3	Inner Time Engine	يحفظ تاريخ التعلم
4	Vascular Compute Router	يوزع الموارد
5	Global Workspace	يجمع السياق النشط
6	Agency Selector	يختار الفعل
7	Perturbation Court	يهدم ويصلح
8	World-Effect Loop	يقيس الأثر
9	Self-Reconstruction Memory	يعيد بناء الذاكرة

```
FailureDiagnosis = locate({extraction,scoring,retrieval,routing,court,execution,memory})
```

نموذج البيانات الموحد

الكائنات التي تحمل الصرح داخل الكود

لكي لا يبقى الصرح لغويا، يحتاج نموذجا موحدا للبيانات. Knowledge Atom يحمل النص والدعوى والمصدر والثقة والخطر؛ Difference Object يحمل الفرق وbaseline وMCD؛ Question Object يحمل السؤال والنية والقيود؛ Action Object يحمل الفعل والكلفة والرجوع؛ World Effect Object يحمل ما تغير قبل/بعد؛ وMemory Patch يحمل ما يجب تحديثه في SRM.

هذا النموذج يضمن أن كل نظرية تملك مقابلا تقنيا. فعندما نقول "السؤال حاكم" يظهر ذلك في Question Object. وعندما نقول "الأثر يعيد بناء الذات" يظهر في WorldEffect وMemoryPatch. وعندما نقول "محكمة الهدم" تظهر في CourtVerdict. بهذا تصبح اللغة متصلة بالكود.

```
class KnowledgeAtom:
    id: str; text: str; source: str
    claim_rank: str; confidence: float
    evidence: list; risks: list; links: list

class DifferenceObject:
    delta: str; baseline: str
    mcd_vector: dict; evidence_links: list
    novelty: float; noise: float
```

قاعدة الكائنات

لا يدخل أي atom إلى HyperKB بلا مصدر أو رتبة أو علاقة بسؤال أو أثر محتمل.

تطبيع المصطلحات: MICAI/MCAI/MICAGI/neoAGI

الخريطة التي تمنع التداخل اللفظي

MICAI هو المظلة العليا: نظام إدماج معرفي-وكالي يربط السؤال، الفرق، الحاكمية، الذاكرة، الهدم، والأثر. MCAI هو نواة الوكالة الميتا-تعقيدية داخل MICAI. MICAGI هو امتداد موجه للذكاء العام يجعل النظام قابلا لتشغيل وكلاء وأدوات ونماذج متعددة. neoAGI هو نمط النشر والهندسة: orchestration، أدوات، ذاكرة، مراقبة، وتغذية راجعة.

بهذا لا تتنافس الأسماء بل تتوزع. إذا كان السؤال حول الفلسفة العامة فنحن في MICAI. إذا كان حول دورة القرار فنحن في MCAI. إذا كان حول قابلية العمومية بين المجالات فنحن في MICAGI. وإذا كان حول التطبيق العملي متعدد النماذج فنحن في neoAGI. هذا التطبيع يحفظ الصرح من التضخم اللفظي.

الاسم	رتبته	المقابل التقني
MICAI	النظام المندمج	Architecture + protocols
MCAI	القلب الوكالي	Agentic cycle
HAKIM-MCD	النواة الوحداتية	Modules/services
QMCD/MCD	رياضيات الفرق والسؤال	Scorers
HyperKB OS	ذاكرة التشغيل	Graph+vector+schemas
MICAGI/neoAGI	النشر العام	Multi-agent orchestration

HyperKnowledge OS

المعرفة كجهاز تشغيل لا أرشيف

HyperKB OS هو الطبقة التي تمنع المشروع من أن يتحول إلى ملفات جميلة منفصلة. المعرفة داخله ليست نصوصا فقط، بل memory patches، tools، tests، loops، skills، objections، evidence، claims. السؤال ليس: أين توجد المعلومة؟ بل: ماذا تفعل المعلومة؟ ما الفرق الذي تكشفه؟ أي مهارة تولد؟ أي محكمة تهدمها؟ أي أثر تقيسه؟

يحتاج HyperKB إلى ثلاث ذاكرات متداخلة: graph للعلاقات السببية والمفهومية، vector للدلالة والاسترجاع، وجداول للقياسات والأثر. الاعتماد على embeddings وحدها يفقد العلاقات، والاعتماد على graph وحده يفقد المرونة الدلالية، والاعتماد على الجداول وحدها يفقد المعنى. الثلاثية تصنع ذاكرة قابلة للتشغيل.

```
Knowledge -> Representation -> Skill -> Loop -> Agent -> Effect -> Reconstruction
```

معيار الجودة

HyperKB لا يقاس بحجم ما يخزن، بل بقدرته على توليد فعل أدق، نقد أقوى، وذاكرة أقل تكرارا.

GraphRAG والذاكرة الثلاثية

لماذا لا يكفي الاسترجاع الدلالي

GraphRAG داخل MICAI ليس مجرد استرجاع، بل استرجاع محكوم بالسؤال والفرق. عندما يأتي سؤال، يبحث النظام عن الأدلة والعلاقات والاعتراضات والفجوات لا عن الجمل المتشابهة فقط. لذلك تحفظ العقد: دعاوى، مفاهيم، أدوات، تجارب، محاكم، أشخاص، فروع، وأثر. وتحفظ الحواف: يسبب، يعترض، يفسر، يشترط، يوسع، يهدم، يولد.

الذاكرة الثلاثية تجعل كل بحث قابلا للتتبع. النص يعطي مادة، graph يعطي علاقة، vector يعطي قربا دلاليا، والجداول تعطي قياسا. وعندما يختلف الثلاثة، ترفع الحاكمية درجة عدم اليقين بدل أن تغطي التناقض. هذه الآلية ضرورية لدراسة إمكانات الذكاء الفتّاح لا لاستهلاك المعرفة فقط.

نوع الذاكرة	تحفظ	تخطر إذا غابت
Graph	العلاقات والسببية	فقدان البنية
Vector	الدلالة والتشابه	فقدان المرونة
Table	القياس والحالات	فقدان التقييم
Trace	سبب الاسترجاع	فقدان الثقة

```
retrieve(q):
    q_obj = build_question(q)
    seeds = vector_search(q_obj)
    graph_pack = expand_by_edges(seeds, edge_types=["evidence","objection","tool"])
    return rank_by_qmcd(graph_pack, q_obj)
```

صفحة 23 / 75

HyperSkills

تحويل المعرفة إلى قدرة قابلة لإعادة الاستخدام

HyperSkill هي وحدة معرفة عملية لها مدخلات، شروط، خطوات، مخرجات، حدود، أخطاء شائعة، ومؤشر نجاح. لا تصبح المعلومة مهارة لأنها جميلة، بل لأنها تكرر النجاح عبر عائلة مهام. لذلك يجب أن تمر كل HyperSkill عبر محكمة: هل هي عامة بما يكفي؟ هل حدودها واضحة؟ هل تحمل أمثلة فشل؟ هل تنتج artifact قابل للفحص؟

في الصرح الجديد، HyperSkills هي طريقة تكثيف كل نظرية في قدرة. MCD ينتج مهارة تقييم الفرق؛ QMCD ينتج مهارة هندسة السؤال؛ HAKIM ينتج مهارة تشخيص الفشل؛ HyperKB ينتج مهارة بناء ذاكرة؛ Court تنتج مهارة هدم؛ WorldEffect ينتج مهارة اختبار. بهذه الطريقة تتحول النظرية إلى تدريب لا إلى حفظ.

خاصية HyperSkill	التفسير
Reusable	تستعمل في أكثر من مهمة
Bounded	تعرف حدودها
Traceable	تملك evidence وlogs
Measurable	لها metric نجاح
Repairable	تعرف كيف تصلح فشلها

قانون الترقية

لا ترقى مهارة إلى HyperSkill إلا بعد نجاح متكرر عبر مهام مختلفة مع انخفاض واضح في الأخطاء أو الزمن أو الضجيج.

صفحة 24 / 75

HyperLoops

من المهارة إلى دورة تعلم ذات ذاكرة

HyperLoop هو workflow ذا ذاكرة يعيد استعمال HyperSkill ضمن حلقة: مدخل، سؤال، فرق، فعل، أثر، إصلاح، ذاكرة. الفرق بين workflow عادي وHyperLoop أن الثاني يتذكر سبب نجاحه وفشله ويحدث نفسه. لذلك يصبح قابلا للتحسن عبر الزمن، لا مجرد مجموعة خطوات ثابتة.

في Hermes/neoAGI يمكن لكل يوم توليد loops جديدة، لكن شرط الجودة أن تكون موسومة، قابلة للاختبار، ومتصلة بالذاكرة. كثرة loops بلا حاكمية تتحول إلى فوضى. لذلك يخضع كل loop إلى MCD/QMCD: ما الفرق الذي يبرر وجوده؟ أي سؤال يخدم؟ ما أثره؟ متى يلغى أو يدمج؟

```
HyperLoop = HyperSkill + State + Feedback + RepairPolicy + MemoryPatch
```

مرحلة	وظيفة
Plan	بناء الفعل
Execute	تشغيل الأداة أو الوكيل
Observe	قياس الأثر
Court	كشف الفشل
Patch	تحديث الذاكرة
Promote/Retire	ترقية أو حذف

Prompt-as-Kernel

البرومبت كواجهة تشغيل لا نص ساحر

في هذا الصرح، البرومبت ليس تعويذة طويلة؛ إنه kernel يحدد الدور، العقد، المدخلات، المخرجات، الحدود، محكمة الهدم، سياسة الأدوات، وقواعد تحديث الذاكرة. لذلك يجب أن يكون versioned، قابل القياس، وقابلا للتفكيك إلى وحدات: System Identity، Task Contract، Context Pack، Tool Policy، Output Schema، Court Checklist، Memory Patch Rules.

كل ميغابرومبت سابق يمكن إعادة بنائه كوحدة OS. بدلا من نص ضخم يرهق النموذج، نملك kernels قابلة للتركيب حسب المهمة. فإذا طلب المستخدم ورقة علمية، يستدعى Kernel البحث؛ وإذا طلب كودا، يستدعى Kernel الأداة؛ وإذا طلب نقدا، يستدعى Kernel المحكمة. هذه modularity تخفض الضجيج وترفع الاتساق.

```
PromptKernel = {
  "role": "research-agent",
  "input_schema": ["question","sources","constraints"],
  "output_schema": ["claims","tools","tests","risks"],
  "court": ["evidence","novelty","operation","effect"],
  "memory_patch": true
}
```

شرط الجودة

كل prompt لا يحدد معيار نجاح ومجال امتناع ومحكمة اعتراض هو نص قابل للانحراف.

Knowledge-to-Artifact Compiler

تحويل النظرية إلى PDF أو كود أو لوحة قياس

Compiler المعرفة إلى artifact هو الآلية التي تمنع الانفصال بين الفكر والإنتاج. يأخذ claims وschemas وquestions وtools وينتج مخرجا: ورقة PDF، ملف Markdown، كود، dashboard، تجربة، أو خطة تنفيذ. لكنه لا يكتفي بالبناء، يرفق verification: هل فتح الملف؟ هل تعمل الدالة؟ هل خلت الصفحات من الخلل؟ هل تتبع المخرج القيود؟

هذا هو قلب العمل المؤسسي لمشروع مساعف: النظرية لا تنتهي عند صياغة جميلة، بل تتحول إلى artifact قابل للاختبار. كل artifact يعود إلى HyperKB مع سجل: المدخلات، التعديلات، الأخطاء، درجة النجاح، وتعليق المحكمة. وبذلك لا تضيع الأعمال السابقة في المحادثات، بل تتحول إلى نظام إنتاج معرفي.

```
Artifact = Compile(KnowledgePack, OutputSchema, StyleRules, VerificationSuite)
```

نوع artifact	اختباره
PDF	عدد الصفحات، سلامة RTL، عدم القص
Code	tests، type check، logs
MD KB	chunk IDs، روابط، schemas
Dashboard	metrics حية، alerts
Prompt	نجاح عبر مهام، انخفاض drift

NeoAGI Orchestration وHermes/Codex

توزيع الذكاء قبل ادعاء امتلاكه

neoAGI Orchestration لا تعني أن الذكاء العام تحقق، بل أن سلوكا عاما يمكن تقريبه عبر توزيع المهام بين نماذج، أدوات، ذاكرة، محاكم، وواجهات تنفيذ. نموذج للبحث، آخر للكود، آخر للتلخيص، آخر للرؤية، وراوتر يختار حسب MCD/QMCD والخطر والتكلفة والتاريخ. هذا يمنع استعمال نموذج ضخم لكل شيء ويمنع أيضا اختيارا اقتصاديا أعمى.

Hermes/Codex داخل الصرح ليس مجرد منصة تنفيذ، بل Command OS: يستقبل mission، يبني context pack، يستدعي hyperloops، ينفذ tools، يسجل الأثر، ويحدث الذاكرة. قوته لا تأتي من الأتمتة وحدها، بل من محكمة السؤال والفرق. لذلك كل automation بلا حاكمية قد يزيد السرعة لكنه لا يزيد الذكاء.

طبقة orchestration	وظيفتها
Model Router	اختيار النموذج
Tool Router	اختيار الأداة
Memory Router	اختيار السياق
Court Router	اختيار عمق الهدم
Cost Router	ضبط الميزانية
Effect Router	تحديد الاختبار

احتراز

المشروع يبني شروط الاقتراب من AGI-oriented behavior، لا يدعي أن توزيع الأدوات وحده يخلق AGI كاملا.

تحليل الفراغات الممكنة

لماذا نحتاج سبع نظريات مساعفية سيدة

بعد تجميع النظريات السابقة تظهر فراغات لا تكفيها الأسماء الموجودة. هناك فراغ في حاكمية السؤال حين يصبح السؤال هو الذي يوزع الموارد، وفراغ في تراكب الوكالة حين توجد مسارات متعددة قبل الفعل، وفراغ في ضغط الفرق بين نماذج متنافسة، وفراغ في ذاكرة تعيد بناء الذات لا تحفظ فقط، وفراغ في تعريف الأثر الوثوقي، وفراغ في جعل المحكمة مولدة لا نافية، وفراغ في نظرية جامعة للذكاء الفتّاح العام.

النظريات السبع الآتية لا تلغي النظريات السابقة، بل تتوجها. كل نظرية سيدة تعمل كقبة فوق مجموعة فرعية: تجمعها، تعطيها قانونا، وتفتح مسار اختبار. لذلك سميت باسم مساعف لا باعتبارها زينة نسبية، بل لأنها تنظم الإمضاء المعرفي للمشروع: السؤال، الفرق، الوعاء، الوكالة، الهدم، الأثر، والذاكرة.

الفراغ	النظرية السيدة
حاكمية السؤال والموارد	نظرية مساعف للحاكمية السؤالانية الوعائية العامة
تعدد الوكالات قبل الفعل	نظرية مساعف للتراكب الوكالي المفتوح
اختلاف النماذج وضغطها	نظرية مساعف للضغط الميتا-اختلافي متعدد النماذج
الذاكرة والهوية	نظرية مساعف للذاكرة الوعائية المعيدة للبناء
الأثر والثقة	نظرية مساعف للأثر العالمي الوثوقي
الهدم المنتج	نظرية مساعف للمحكمة التوليدية الهدامة
الجامعة العليا	نظرية مساعف للذكاء الفتّاح العام

النظرية السيدة 1: نظرية مساعف للحاكمية السؤالانية الوعائية العامة

Moussaif General Vascular Question Governance Theory

تقرر أن السؤال ليس مدخلا لغويا، بل حاكمية توزع الموارد عبر أوعية النظام. كل سؤال يملك قدرة فتح، قدرة ضبط، قدرة هدم، وقدرة إنزال. لذلك لا يحق للنظام أن يعامل كل الأسئلة بدرجة واحدة؛ بل يحدد وزن السؤال ثم يفتح الفروق التي تستحق الحساب.

تعمل هذه النظرية كقبة فوق عائلة من المفاهيم السابقة، ولا تستبدلها. وظيفتها أن تجعل ما كان موزعا بين السؤال والفرق والوكالة أو الذاكرة قانونا واحدا قابلا للإنزال. معيار قبولها أن يظهر لها مقابل تقني: كائن بيانات، دالة قياس، وحدة تشغيل، محكمة، ومؤشر أثر.

```
QG(q)=Clarity+Scope+Generativity+Falsifiability+Actionability-Drift-Ambiguity
```

الفراغ الذي تسده

تعالج فراغا مركزيا: من يحكم الراوتر قبل أن يقيس الفرق؟ إنها تجعل السؤال نفسه كائنا تشغيليا له score، وبهذا تغلق الفجوة بين prompt engineering وepistemic governance.

صفحة 30 / 75

اختبار النظرية السيدة 1

آلية التشغيل والهدم: نظرية مساعف للحاكمية السؤالانية الوعائية العامة

لإدخال هذه النظرية في الصرح، تعامل كحزمة تشغيل: schema يحمل مدخلاتها، scorer يقدر وزنها، policy يحدد متى تستعمل، court يهاجمها، وbenchmark يختبر أثرها. لا تكفي قوة التسمية؛ يجب أن نعرف كيف تفشل وكيف تصلح فشلها.

مستوى	اختبار
تعريف	هل يميز المفهوم عن سابقه؟
آلية	هل يصف إدخال/تحويل/إخراج؟
قياس	هل ينتج متجها أو مؤشرا؟
تنفيذ	هل يمكن كتابة module له؟
هدم	ما الاعتراض الفاصل؟
أثر	هل حسّن قرارا أو ذاكرة أو أداة؟

```
def test_theory_1(case):
    model = build_operational_schema(case)
    score = measure_core_effect(model)
    court = run_demolition_court(model)
    return {"score": score, "repairs": court.repairs,
            "accepted": score > threshold and not court.fatal}
```

رتبة الادعاء

النظرية مقترحة كإطار بحثي قابل للقياس. لا ترتفع إلى رتبة نظرية مثبتة إلا بعد تجارب مقارنة، ablations، ومقاييس أثر متكررة.

صفحة 31 / 75

النظرية السيدة 2: نظرية مساعف للتراكب الوكالي المفتوح

Moussaif Open Agentic Superposition Theory

تقول إن الوكيل قبل الفعل يعيش في تراكب مسارات: بحث، كتابة، كود، امتناع، سؤال، تجربة، أو محكمة. لا ينهار هذا التراكب إلا عبر حاكمية تجمع القيمة والخطر والرجوع والأثر.

تعمل هذه النظرية كقبة فوق عائلة من المفاهيم السابقة، ولا تستبدلها. وظيفتها أن تجعل ما كان موزعا بين السؤال والفرق والوكالة أو الذاكرة قانونا واحدا قابلا للإنزال. معيار قبولها أن يظهر لها مقابل تقني: كائن بيانات، دالة قياس، وحدة تشغيل، محكمة، ومؤشر أثر.

```
π(a|s)=Softmax(Value(a)+Evidence(a)+Reversibility(a)-Risk(a)-Cost(a))
```

الفراغ الذي تسده

تعالج فراغ الاختيار داخل agentic systems: ليست الوكالة فعلا مباشرا، بل إدارة بدائل قبل الحسم.

اختبار النظرية السيدة 2

آلية التشغيل والهدم: نظرية مساعف للتراكب الوكالي المفتوح

لإدخال هذه النظرية في الصرح، تعامل كحزمة تشغيل: schema يحمل مدخلاتها، scorer يقدر وزنها، policy يحدد متى تستعمل، court يهاجمها، وbenchmark يختبر أثرها. لا تكفي قوة التسمية؛ يجب أن نعرف كيف تفشل وكيف تصلح فشلها.

مستوى	اختبار
تعريف	هل يميز المفهوم عن سابقه؟
آلية	هل يصف إدخال/تحويل/إخراج؟
قياس	هل ينتج متجها أو مؤشرا؟
تنفيذ	هل يمكن كتابة module له؟
هدم	ما الاعتراض الفاصل؟
أثر	هل حسّن قرارا أو ذاكرة أو أداة؟

```
def test_theory_2(case):
    model = build_operational_schema(case)
    score = measure_core_effect(model)
    court = run_demolition_court(model)
    return {"score": score, "repairs": court.repairs,
            "accepted": score > threshold and not court.fatal}
```

رتبة الادعاء

النظرية مقترحة كإطار بحثي قابل للقياس. لا ترتفع إلى رتبة نظرية مثبتة إلا بعد تجارب مقارنة، ablations، ومقاييس أثر متكررة.

النظرية السيدة 3: نظرية مساعف للضغط الميتا-اختلافي متعدد النماذج

Moussaif Multi-Model Meta-Difference Compression Theory

تجعل اختلاف النماذج مادة معرفة. عندما تختلف النماذج لا نختار فورا، بل نستخرج الفرق بين تفسيرها، ثم نضغط الفرق إلى قاعدة أعلى أو سؤال فاصل.

تعمل هذه النظرية كقبة فوق عائلة من المفاهيم السابقة، ولا تستبدلها. وظيفتها أن تجعل ما كان موزعا بين السؤال والفرق والوكالة أو الذاكرة قانونا واحدا قابلا للإنزال. معيار قبولها أن يظهر لها مقابل تقني: كائن بيانات، دالة قياس، وحدة تشغيل، محكمة، ومؤشر أثر.

```
MMC = Compress({Δ_m1, Δ_m2, ... Δ_mn}) - LostCriticalDifferences
```

الفراغ الذي تسده

تعالج فراغ orchestration المعرفي: كيف نستفيد من تعدد النماذج دون أن نحوله إلى تصويت أعمى.

اختبار النظرية السيدة 3

آلية التشغيل والهدم: نظرية مساعف للضغط الميتا-اختلافي متعدد النماذج

لإدخال هذه النظرية في الصرح، تعامل كحزمة تشغيل: schema يحمل مدخلاتها، scorer يقدر وزنها، policy يحدد متى تستعمل، court يهاجمها، وbenchmark يختبر أثرها. لا تكفي قوة التسمية؛ يجب أن نعرف كيف تفشل وكيف تصلح فشلها.

مستوى	اختبار
تعريف	هل يميز المفهوم عن سابقه؟
آلية	هل يصف إدخال/تحويل/إخراج؟
قياس	هل ينتج متجها أو مؤشرا؟
تنفيذ	هل يمكن كتابة module له؟
هدم	ما الاعتراض الفاصل؟
أثر	هل حسّن قرارا أو ذاكرة أو أداة؟

```
def test_theory_3(case):
    model = build_operational_schema(case)
    score = measure_core_effect(model)
    court = run_demolition_court(model)
    return {"score": score, "repairs": court.repairs,
            "accepted": score > threshold and not court.fatal}
```

رتبة الادعاء

النظرية مقترحة كإطار بحثي قابل للقياس. لا ترتفع إلى رتبة نظرية مثبتة إلا بعد تجارب مقارنة، ablations، ومقاييس أثر متكررة.

النظرية السيدة 4: نظرية مساعف للذاكرة الوعائية المعيدة للبناء

Moussaif Vascular Self-Reconstructing Memory Theory

الذاكرة ليست تخزينا، بل إعادة توجيه للذات بعد كل أثر. تحفظ ما تغير في السؤال والفرق والقرار والنتيجة والحدود، ثم تعيد تشكيل سياسات النظام.

تعمل هذه النظرية كقبة فوق عائلة من المفاهيم السابقة، ولا تستبدلها. وظيفتها أن تجعل ما كان موزعا بين السؤال والفرق والوكالة أو الذاكرة قانونا واحدا قابلا للإنزال. معيار قبولها أن يظهر لها مقابل تقني: كائن بيانات، دالة قياس، وحدة تشغيل، محكمة، ومؤشر أثر.

```
SRM_{t+1}=SRM_t + Patch(WE,Court,UserFeedback,NewEvidence) - ObsoleteNoise
```

الفراغ الذي تسده

تعالج فراغ الاستمرارية: كيف لا يبدأ النظام من الصفر ولا يكرر الأخطاء ولا يبقى سجلا ميتا.

اختبار النظرية السيدة 4

آلية التشغيل والهدم: نظرية مساعف للذاكرة الوعائية المعيدة للبناء

لإدخال هذه النظرية في الصرح، تعامل كحزمة تشغيل: schema يحمل مدخلاتها، scorer يقدر وزنها، policy يحدد متى تستعمل، court يهاجمها، وbenchmark يختبر أثرها. لا تكفي قوة التسمية؛ يجب أن نعرف كيف تفشل وكيف تصلح فشلها.

مستوى	اختبار
تعريف	هل يميز المفهوم عن سابقه؟
آلية	هل يصف إدخال/تحويل/إخراج؟
قياس	هل ينتج متجها أو مؤشرا؟
تنفيذ	هل يمكن كتابة module له؟
هدم	ما الاعتراض الفاصل؟
أثر	هل حسّن قرارا أو ذاكرة أو أداة؟

```
def test_theory_4(case):
    model = build_operational_schema(case)
    score = measure_core_effect(model)
    court = run_demolition_court(model)
    return {"score": score, "repairs": court.repairs,
            "accepted": score > threshold and not court.fatal}
```

رتبة الادعاء

النظرية مقترحة كإطار بحثي قابل للقياس. لا ترتفع إلى رتبة نظرية مثبتة إلا بعد تجارب مقارنة، ablations، ومقاييس أثر متكررة.

النظرية السيدة 5: نظرية مساعف للأثر العالمي الوثوقي

Moussaif Reliable World-Effect Theory

تعرف الذكاء بقدرته على إنتاج أثر قابل للقياس والتحقق والرجوع، لا ببلاغة output. الأثر الوثوقي يجمع delta قبل/بعد، الثقة، الاستمرارية، وانخفاض الآثار الجانبية.

تعمل هذه النظرية كقبة فوق عائلة من المفاهيم السابقة، ولا تستبدلها. وظيفتها أن تجعل ما كان موزعا بين السؤال والفرق والوكالة أو الذاكرة قانونا واحدا قابلا للإنزال. معيار قبولها أن يظهر لها مقابل تقني: كائن بيانات، دالة قياس، وحدة تشغيل، محكمة، ومؤشر أثر.

```
RWE = ΔObserved × Persistence × Confidence × Reversibility - SideEffects - UnverifiedClaims
```

الفراغ الذي تسده

تعالج فراغ العلاقة بالعالم: متى نقول إن المخرج لم يكن جميلا فقط بل أحدث تحسنا صالحا.

اختبار النظرية السيدة 5

آلية التشغيل والهدم: نظرية مساعف للأثر العالمي الوثوقي

لإدخال هذه النظرية في الصرح، تعامل كحزمة تشغيل: schema يحمل مدخلاتها، scorer يقدر وزنها، policy يحدد متى تستعمل، court يهاجمها، وbenchmark يختبر أثرها. لا تكفي قوة التسمية؛ يجب أن نعرف كيف تفشل وكيف تصلح فشلها.

مستوى	اختبار
تعريف	هل يميز المفهوم عن سابقه؟
آلية	هل يصف إدخال/تحويل/إخراج؟
قياس	هل ينتج متجها أو مؤشرا؟
تنفيذ	هل يمكن كتابة module له؟
هدم	ما الاعتراض الفاصل؟
أثر	هل حسّن قرارا أو ذاكرة أو أداة؟

```
def test_theory_5(case):
    model = build_operational_schema(case)
    score = measure_core_effect(model)
    court = run_demolition_court(model)
    return {"score": score, "repairs": court.repairs,
            "accepted": score > threshold and not court.fatal}
```

رتبة الادعاء

النظرية مقترحة كإطار بحثي قابل للقياس. لا ترتفع إلى رتبة نظرية مثبتة إلا بعد تجارب مقارنة، ablations، ومقاييس أثر متكررة.

النظرية السيدة 6: نظرية مساعف للمحكمة التوليدية الهدامة

Moussaif Generative Destruction Court Theory

تجعل الهدم مولدا: كل اعتراض يجب أن ينتج إصلاحا أو اختبارا أو قيدا أو سؤالا أعلى. المحكمة لا تقول لا فقط؛ بل تعيد تشكيل الدعوى لتصبح أصلب.

تعمل هذه النظرية كقبة فوق عائلة من المفاهيم السابقة، ولا تستبدلها. وظيفتها أن تجعل ما كان موزعا بين السؤال والفرق والوكالة أو الذاكرة قانونا واحدا قابلا للإنزال. معيار قبولها أن يظهر لها مقابل تقني: كائن بيانات، دالة قياس، وحدة تشغيل، محكمة، ومؤشر أثر.

```
GDC(output)=Attack(output)->BrokenClaims->Repairs->Tests->ResidualRisks
```

الفراغ الذي تسده

تعالج فراغ النقد العدمي: كيف نمنع التساهل دون أن نقتل البناء.

اختبار النظرية السيدة 6

آلية التشغيل والهدم: نظرية مساعف للمحكمة التوليدية الهدامة

لإدخال هذه النظرية في الصرح، تعامل كحزمة تشغيل: schema يحمل مدخلاتها، scorer يقدر وزنها، policy يحدد متى تستعمل، court يهاجمها، وbenchmark يختبر أثرها. لا تكفي قوة التسمية؛ يجب أن نعرف كيف تفشل وكيف تصلح فشلها.

مستوى	اختبار
تعريف	هل يميز المفهوم عن سابقه؟
آلية	هل يصف إدخال/تحويل/إخراج؟
قياس	هل ينتج متجها أو مؤشرا؟
تنفيذ	هل يمكن كتابة module له؟
هدم	ما الاعتراض الفاصل؟
أثر	هل حسّن قرارا أو ذاكرة أو أداة؟

```
def test_theory_6(case):
    model = build_operational_schema(case)
    score = measure_core_effect(model)
    court = run_demolition_court(model)
    return {"score": score, "repairs": court.repairs,
            "accepted": score > threshold and not court.fatal}
```

رتبة الادعاء

النظرية مقترحة كإطار بحثي قابل للقياس. لا ترتفع إلى رتبة نظرية مثبتة إلا بعد تجارب مقارنة، ablations، ومقاييس أثر متكررة.

النظرية السيدة 7: نظرية مساعف للذكاء الفتّاح العام الوعائي السؤالي الوكيلي

Moussaif General Fattah Vascular Question-Agentic Intelligence Theory

النظرية الجامعة: الذكاء العام الفتّاح هو قدرة نظام على فتح المسائل المغلقة، التقاط الفروق الفاعلة، توجيه موارده وعائيا، تشغيل وكلاء وأدوات، هدم مخرجاته، اختبار أثره، وإعادة بناء ذاكرته.

تعمل هذه النظرية كقبة فوق عائلة من المفاهيم السابقة، ولا تستبدلها. وظيفتها أن تجعل ما كان موزعا بين السؤال والفرق والوكالة أو الذاكرة قانونا واحدا قابلا للإنزال. معيار قبولها أن يظهر لها مقابل تقني: كائن بيانات، دالة قياس، وحدة تشغيل، محكمة، ومؤشر أثر.

$$MGFI = \int(QG \times MCD \times VCR \times HAKIM \times HKOS \times GDC \times RWE \times SRM)\, dt$$

الفراغ الذي تسده

تعالج الفراغ الأعلى: ما تعريف البرنامج البحثي كله حين نريد الاقتراب من ذكاء عام موثوق لا إجابة عامة فقط.

اختبار النظرية السيدة 7

آلية التشغيل والهدم: نظرية مساعف للذكاء الفتّاح العام الوعائي السؤالي الوكيلي

لإدخال هذه النظرية في الصرح، تعامل كحزمة تشغيل: schema يحمل مدخلاتها، scorer يقدر وزنها، policy يحدد متى تستعمل، court يهاجمها، وbenchmark يختبر أثرها. لا تكفي قوة التسمية؛ يجب أن نعرف كيف تفشل وكيف تصلح فشلها.

مستوى	اختبار
تعريف	هل يميز المفهوم عن سابقه؟
آلية	هل يصف إدخال/تحويل/إخراج؟
قياس	هل ينتج متجها أو مؤشرا؟
تنفيذ	هل يمكن كتابة module له؟
هدم	ما الاعتراض الفاصل؟
أثر	هل حسّن قرارا أو ذاكرة أو أداة؟

```
def test_theory_7(case):
    model = build_operational_schema(case)
    score = measure_core_effect(model)
    court = run_demolition_court(model)
    return {"score": score, "repairs": court.repairs,
            "accepted": score > threshold and not court.fatal}
```

رتبة الادعاء

النظرية مقترحة كإطار بحثي قابل للقياس. لا ترتفع إلى رتبة نظرية مثبتة إلا بعد تجارب مقارنة، ablations، ومقاييس أثر متكررة.

صفحة 43 / 75

الأداة 1: MCD Vectorizer

أداة رياضية-تكنولوجية-كودية داخل الصرح

تحويل Difference Object إلى متجه D/I/T/A/P/C/N مع trace يشرح السبب. هذه الأداة لا توضع كإضافة جانبية، بل كعقدة تشغيل في دورة MICAI. لها inputs واضحة، outputs قابلة للتخزين، metric للحكم، failure mode، وpath repair. قيمتها في أنها تجعل المفهوم قابلا للبرمجة والقياس لا مجرد شرح.

```
score = wD*D+wI*I+wT*T+wA*A+wP*P+wC*C-wN*N
```

المكون	التحديد
المدخلات	Question/Atom/Delta/Context بحسب الأداة
التحويل	دالة تقييم أو routing أو extraction
المخرجات	Object أو score أو policy أو artifact
الفشل المحتمل	ضجيج، انحياز، over-routing، أو فقد دليل
الإصلاح	تعديل أوزان، محكمة، أو تحديث schema

```
# Tool 1: MCD Vectorizer
def run_tool_1(context):
    inputs = validate_inputs(context)
    result = transform(inputs)
    verdict = lightweight_court(result)
    return store_trace(result, verdict)
```

الأداة 2: QMCD Question Gate

أداة رياضية-تكنولوجية-كودية داخل الصرح

قياس جودة السؤال وربط كل فرق بسؤال حاكم وفرعي وهادم وإنزالي. هذه الأداة لا توضع كإضافة جانبية، بل كعقدة تشغيل في دورة MICAI. لها inputs واضحة، outputs قابلة للتخزين، metric للحكم، failure mode، وpath repair. قيمتها في أنها تجعل المفهوم قابلا للبرمجة والقياس لا مجرد شرح.

```
qmcd = mcd * gate(q) * rel(delta,q) - drift(delta,q)
```

المكون	التحديد
المدخلات	Question/Atom/Delta/Context بحسب الأداة
التحويل	دالة تقييم أو routing أو extraction
المخرجات	Object أو score أو policy أو artifact
الفشل المحتمل	ضجيج، انحياز، over-routing، أو فقد دليل
الإصلاح	تعديل أوزان، محكمة، أو تحديث schema

```
# Tool 2: QMCD Question Gate
def run_tool_2(context):
    inputs = validate_inputs(context)
    result = transform(inputs)
    verdict = lightweight_court(result)
    return store_trace(result, verdict)
```

صفحة 45 / 75

الأداة 3: Vascular Compute Router

أداة رياضية-تكنولوجية-كودية داخل الصرح

اختيار النموذج والأداة وعمق المحكمة وميزانية tokens حسب التعقيد والخطر. هذه الأداة لا توضع كإضافة جانبية، بل كعقدة تشغيل في دورة MICAI. لها inputs واضحة، outputs قابلة للتخزين، metric للحكم، failure mode، وpath repair. قيمتها في أنها تجعل المفهوم قابلا للبرمجة والقياس لا مجرد شرح.

```
policy = argmax(expected_effect - cost - latency_risk)
```

المكون	التحديد
المدخلات	Question/Atom/Delta/Context بحسب الأداة
التحويل	دالة تقييم أو routing أو extraction
المخرجات	Object أو score أو policy أو artifact
الفشل المحتمل	ضجيج، انحياز، over-routing، أو فقد دليل
الإصلاح	تعديل أوزان، محكمة، أو تحديث schema

```
# Tool 3: Vascular Compute Router
def run_tool_3(context):
    inputs = validate_inputs(context)
    result = transform(inputs)
    verdict = lightweight_court(result)
    return store_trace(result, verdict)
```

صفحة 46 / 75

الأداة 4: Active Difference Extractor

أداة رياضية-تكنولوجية-كودية داخل الصرح

استخراج الفروق الفاعلة من النصوص والملفات والكود لا مجرد تلخيصها. هذه الأداة لا توضع كإضافة جانبية، بل كعقدة تشغيل في دورة MICAI. لها inputs واضحة، outputs قابلة للتخزين، metric للحكم، failure mode، وpath repair. قيمتها في أنها تجعل المفهوم قابلا للبرمجة والقياس لا مجرد شرح.

```
delta = detect_change(atom, baseline)
```

المكون	التحديد
المدخلات	Question/Atom/Delta/Context بحسب الأداة
التحويل	دالة تقييم أو routing أو extraction
المخرجات	Object أو score أو policy أو artifact
الفشل المحتمل	ضجيج، انحياز، over-routing، أو فقد دليل
الإصلاح	تعديل أوزان، محكمة، أو تحديث schema

```
# Tool 4: Active Difference Extractor
def run_tool_4(context):
    inputs = validate_inputs(context)
    result = transform(inputs)
    verdict = lightweight_court(result)
    return store_trace(result, verdict)
```

الأداة 5: Inner Time Scheduler

أداة رياضية-تكنولوجية-كودية داخل الصرح

تحديد عدد دورات التفكير والتحقق بناء على الخطر وفجوة الدليل. هذه الأداة لا توضع كإضافة جانبية، بل كعقدة تشغيل في دورة MICAI. لها inputs واضحة، outputs قابلة للتخزين، metric للحكم، failure mode، وpath repair. قيمتها في أنها تجعل المفهوم قابلا للبرمجة والقياس لا مجرد شرح.

```
depth = ceil(alpha*risk + beta*mcd + gamma*gap)
```

المكون	التحديد
المدخلات	Question/Atom/Delta/Context بحسب الأداة
التحويل	دالة تقييم أو routing أو extraction
المخرجات	Object أو score أو policy أو artifact
الفشل المحتمل	ضجيج، انحياز، over-routing، أو فقد دليل
الإصلاح	تعديل أوزان، محكمة، أو تحديث schema

```
# Tool 5: Inner Time Scheduler
def run_tool_5(context):
    inputs = validate_inputs(context)
    result = transform(inputs)
    verdict = lightweight_court(result)
    return store_trace(result, verdict)
```

صفحة 48 / 75

الأداة 6: HyperKB Graph Schema

أداة رياضية-تكنولوجية-كودية داخل الصرح

بناء graph/vector/tables يحفظ الدعوى والدليل والاعتراض والأثر. هذه الأداة لا توضع كإضافة جانبية، بل كعقدة تشغيل في دورة MICAI. لها inputs واضحة، outputs قابلة للتخزين، metric للحكم، failure mode، وpath repair. قيمتها في أنها تجعل المفهوم قابلا للبرمجة والقياس لا مجرد شرح.

```
node = {claim, evidence, objections, tools, effects}
```

المكون	التحديد
المدخلات	Question/Atom/Delta/Context بحسب الأداة
التحويل	دالة تقييم أو routing أو extraction
المخرجات	Object أو score أو policy أو artifact
الفشل المحتمل	ضجيج، انحياز، over-routing، أو فقد دليل
الإصلاح	تعديل أوزان، محكمة، أو تحديث schema

```
# Tool 6: HyperKB Graph Schema
def run_tool_6(context):
    inputs = validate_inputs(context)
    result = transform(inputs)
    verdict = lightweight_court(result)
    return store_trace(result, verdict)
```

صفحة 49 / 75

الأداة 7: Perturbation Court Engine

أداة رياضية-تكنولوجية-كودية داخل الصرح

توليد اضطرابات على السؤال والافتراض لتعريض المخرج للكسر ثم الإصلاح. هذه الأداة لا توضع كإضافة جانبية، بل كعقدة تشغيل في دورة MICAI. لها inputs واضحة، outputs قابلة للتخزين، metric للحكم، failure mode، وpath repair. قيمتها في أنها تجعل المفهوم قابلا للبرمجة والقياس لا مجرد شرح.

```
verdict = court.attack(output, perturbations)
```

المكون	التحديد
المدخلات	Question/Atom/Delta/Context بحسب الأداة
التحويل	دالة تقييم أو routing أو extraction
المخرجات	Object أو score أو policy أو artifact
الفشل المحتمل	ضجيج، انحياز، over-routing، أو فقد دليل
الإصلاح	تعديل أوزان، محكمة، أو تحديث schema

```
# Tool 7: Perturbation Court Engine
def run_tool_7(context):
    inputs = validate_inputs(context)
    result = transform(inputs)
    verdict = lightweight_court(result)
    return store_trace(result, verdict)
```

صفحة 50 / 75

الأداة 8: World-Effect Evaluator

أداة رياضية-تكنولوجية-كودية داخل الصرح

قياس ما تغير قبل/بعد في artifact أو قرار أو ذاكرة أو تجربة. هذه الأداة لا توضع كإضافة جانبية، بل كعقدة تشغيل في دورة MICAI. لها inputs واضحة، outputs قابلة للتخزين، metric للحكم، failure mode، وpath repair. قيمتها في أنها تجعل المفهوم قابلا للبرمجة والقياس لا مجرد شرح.

```
effect = observed_delta * confidence - side_effects
```

المكون	التحديد
المدخلات	Question/Atom/Delta/Context بحسب الأداة
التحويل	دالة تقييم أو routing أو extraction
المخرجات	Object أو score أو policy أو artifact
الفشل المحتمل	ضجيج، انحياز، over-routing، أو فقد دليل
الإصلاح	تعديل أوزان، محكمة، أو تحديث schema

```
# Tool 8: World-Effect Evaluator
def run_tool_8(context):
    inputs = validate_inputs(context)
    result = transform(inputs)
    verdict = lightweight_court(result)
    return store_trace(result, verdict)
```

صفحة 51 / 75

الأداة 9: Self-Reconstruction Memory Compiler

أداة رياضية-تكنولوجية-كودية داخل الصرح

تحويل الأثر والحكم إلى patches في الذاكرة والسياسات. هذه الأداة لا توضع كإضافة جانبية، بل كعقدة تشغيل في دورة MICAI. لها inputs واضحة، outputs قابلة للتخزين، metric للحكم، failure mode، وpath repair. قيمتها في أنها تجعل المفهوم قابلا للبرمجة والقياس لا مجرد شرح.

```
memory.apply(patch_from(effect,court))
```

المكون	التحديد
المدخلات	Question/Atom/Delta/Context بحسب الأداة
التحويل	دالة تقييم أو routing أو extraction
المخرجات	Object أو score أو policy أو artifact
الفشل المحتمل	ضجيج، انحياز، over-routing، أو فقد دليل
الإصلاح	تعديل أوزان، محكمة، أو تحديث schema

```python
# Tool 9: Self-Reconstruction Memory Compiler
def run_tool_9(context):
    inputs = validate_inputs(context)
    result = transform(inputs)
    verdict = lightweight_court(result)
    return store_trace(result, verdict)
```

صفحة 52 / 75

الأداة 10: Multi-Model Orchestration Router

أداة رياضية-تكنولوجية-كودية داخل الصرح

اختيار النماذج المتخصصة وتوليف نتائجها وفق QMCD لا التصويت الساذج. هذه الأداة لا توضع كإضافة جانبية، بل كعقدة تشغيل في دورة MICAI. لها inputs واضحة، outputs قابلة للتخزين، metric للحكم، failure mode، وpath repair. قيمتها في أنها تجعل المفهوم قابلا للبرمجة والقياس لا مجرد شرح.

```
winner = select_by_qmcd(candidates)
```

المكون	التحديد
المدخلات	Question/Atom/Delta/Context بحسب الأداة
التحويل	دالة تقييم أو routing أو extraction
المخرجات	Object أو score أو policy أو artifact
الفشل المحتمل	ضجيج، انحياز، over-routing، أو فقد دليل
الإصلاح	تعديل أوزان، محكمة، أو تحديث schema

```
# Tool 10: Multi-Model Orchestration Router
def run_tool_10(context):
    inputs = validate_inputs(context)
    result = transform(inputs)
    verdict = lightweight_court(result)
    return store_trace(result, verdict)
```

صفحة 53 / 75

الأداة 11: Knowledge-to-Artifact Compiler

أداة رياضية-تكنولوجية-كودية داخل الصرح

تحويل حزمة معرفة إلى PDF/MD/Code/Dashboard مع تحقق آلي. هذه الأداة لا توضع كإضافة جانبية، بل كعقدة تشغيل في دورة MICAI. لها inputs واضحة، outputs قابلة للتخزين، metric للحكم، failure mode، وpath repair. قيمتها في أنها تجعل المفهوم قابلا للبرمجة والقياس لا مجرد شرح.

```
artifact = compile(pack, schema, verifier)
```

المكون	التحديد
المدخلات	Question/Atom/Delta/Context بحسب الأداة
التحويل	دالة تقييم أو routing أو extraction
المخرجات	Object أو score أو policy أو artifact
الفشل المحتمل	ضجيج، انحياز، over-routing، أو فقد دليل
الإصلاح	تعديل أوزان، محكمة، أو تحديث schema

```
# Tool 11: Knowledge-to-Artifact Compiler
def run_tool_11(context):
    inputs = validate_inputs(context)
    result = transform(inputs)
    verdict = lightweight_court(result)
    return store_trace(result, verdict)
```

الأداة 12: HyperSkill/HyperLoop Generator

أداة رياضية-تكنولوجية-كودية داخل الصرح

ترقية الخطوات المتكررة إلى مهارات وحلقات بذاكرة. هذه الأداة لا توضع كإضافة جانبية، بل كعقدة تشغيل في دورة MICAI. لها inputs واضحة، outputs قابلة للتخزين، metric للحكم، failure mode، وpath repair. قيمتها في أنها تجعل المفهوم قابلا للبرمجة والقياس لا مجرد شرح.

```
loop = promote(skill, wins, failures)
```

المكون	التحديد
المدخلات	Question/Atom/Delta/Context بحسب الأداة
التحويل	دالة تقييم أو routing أو extraction
المخرجات	Object أو score أو policy أو artifact
الفشل المحتمل	ضجيج، انحياز، over-routing، أو فقد دليل
الإصلاح	تعديل أوزان، محكمة، أو تحديث schema

```
# Tool 12: HyperSkill/HyperLoop Generator
def run_tool_12(context):
    inputs = validate_inputs(context)
    result = transform(inputs)
    verdict = lightweight_court(result)
    return store_trace(result, verdict)
```

صفحة 55 / 75

الأداة 13: Reliability Dashboard

أداة رياضية-تكنولوجية-كودية داخل الصرح

مراقبة الهلوسة والتكلفة والتحسن بعد الهدم ونجاح الأثر. هذه الأداة لا توضع كإضافة جانبية، بل كعقدة تشغيل في دورة MICAI. لها inputs واضحة، outputs قابلة للتخزين، metric للحكم، failure mode، وpath repair. قيمتها في أنها تجعل المفهوم قابلا للبرمجة والقياس لا مجرد شرح.

```
dashboard.log(metrics)
```

المكون	التحديد
المدخلات	Question/Atom/Delta/Context بحسب الأداة
التحويل	دالة تقييم أو routing أو extraction
المخرجات	Object أو score أو policy أو artifact
الفشل المحتمل	ضجيج، انحياز، over-routing، أو فقد دليل
الإصلاح	تعديل أوزان، محكمة، أو تحديث schema

```
# Tool 13: Reliability Dashboard
def run_tool_13(context):
    inputs = validate_inputs(context)
    result = transform(inputs)
    verdict = lightweight_court(result)
    return store_trace(result, verdict)
```

صفحة 56 / 75

الأداة 14: MICAI Simulation Bench

أداة رياضية-تكنولوجية-كودية داخل الصرح

بيئة تجارب تقيس MCD/QMCD/HAKIM/HyperKB عبر حالات معيارية. هذه الأداة لا توضع كإضافة جانبية، بل كعقدة تشغيل في دورة MICAI. لها inputs واضحة، outputs قابلة للتخزين، metric للحكم، failure mode، وpath repair. قيمتها في أنها تجعل المفهوم قابلا للبرمجة والقياس لا مجرد شرح.

```
bench.run(cases, ablations=True)
```

المكون	التحديد
المدخلات	Question/Atom/Delta/Context بحسب الأداة
التحويل	دالة تقييم أو routing أو extraction
المخرجات	Object أو score أو policy أو artifact
الفشل المحتمل	ضجيج، انحياز، over-routing، أو فقد دليل
الإصلاح	تعديل أوزان، محكمة، أو تحديث schema

```
# Tool 14: MICAI Simulation Bench
def run_tool_14(context):
    inputs = validate_inputs(context)
    result = transform(inputs)
    verdict = lightweight_court(result)
    return store_trace(result, verdict)
```

تقنية التدريب 1: منهاج السؤال الحاكم

مسار متقدم للاقتراب من الذكاء الفتّاح العام الموثوق

يدرب النظام على إعادة بناء السؤال قبل الجواب: خام، دقيق، حاكم، هادم، إنزالي، وفوقي. لا تقدم هذه التقنية ضمانا ميتافيزيقيا ببلوغ AGI، لكنها تضمن شرطا بحثيا ضروريا: كل تقدم يصبح مقيسا، قابلا للهدم، وقابلا لإعادة البناء. لذلك يصبح الوصول مسارا منظما لا شعارا.

عنصر التدريب	تنزيله العملي
المادة	حالات متنوعة عالية MCD
المهمة	استخراج/قياس/فعل/هدم/أثر
المكافأة	تحسن مقاس لا بلاغة فقط
العقوبة	ضجيج، overclaim، drift
الترقية	HyperSkill أو policy جديدة بعد تكرار النجاح

```
Train_1 = Data(high_MCD) + Task(QMCD) + Court + EffectReward + MemoryPatch
```

معيار النجاح

تنخفض إعادة العمل، تزيد دقة السؤال، يتحسن اختيار الموارد، ويصبح أثر المخرجات قابلا للملاحظة عبر تجارب متكررة.

تقنية التدريب 2: تدريب الفرق الفاعل

مسار متقدم للاقتراب من الذكاء الفتّاح العام الموثوق

يعطي النظام مواد متشابهة ظاهريا ويكافئه عند التقاط الفرق الذي يغير الحكم أو الفعل. لا تقدم هذه التقنية ضمانا ميتافيزيقيا ببلوغ AGI، لكنها تضمن شرطا بحثيا ضروريا: كل تقدم يصبح مقيسا، قابلا للهدم، وقابلا لإعادة البناء. لذلك يصبح الوصول مسارا منظما لا شعارا.

عنصر التدريب	تنزيله العملي
المادة	حالات متنوعة عالية MCD
المهمة	استخراج/قياس/فعل/هدم/أثر
المكافأة	تحسن مقاس لا بلاغة فقط
العقوبة	ضجيج، overclaim، drift
الترقية	HyperSkill أو policy جديدة بعد تكرار النجاح

```
Train_2 = Data(high_MCD) + Task(QMCD) + Court + EffectReward + MemoryPatch
```

معيار النجاح

تنخفض إعادة العمل، تزيد دقة السؤال، يتحسن اختيار الموارد، ويصبح أثر المخرجات قابلا للملاحظة عبر تجارب متكررة.

تقنية التدريب 3: تدريب محكمة الخصم القوي

مسار متقدم للاقتراب من الذكاء الفتّاح العام الموثوق

يجعل كل مخرج يواجه خصما يحاول إسقاطه بالدليل والمنطق والتنفيذ والسلامة. لا تقدم هذه التقنية ضمانا ميتافيزيقيا ببلوغ AGI، لكنها تضمن شرطا بحثيا ضروريا: كل تقدم يصبح مقيسا، قابلا للهدم، وقابلا لإعادة البناء. لذلك يصبح الوصول مسارا منظما لا شعارا.

عنصر التدريب	تنزيله العملي
المادة	حالات متنوعة عالية MCD
المهمة	استخراج/قياس/فعل/هدم/أثر
المكافأة	تحسن مقاس لا بلاغة فقط
العقوبة	ضجيج، overclaim، drift
الترقية	HyperSkill أو policy جديدة بعد تكرار النجاح

```
Train_3 = Data(high_MCD) + Task(QMCD) + Court + EffectReward + MemoryPatch
```

معيار النجاح

تنخفض إعادة العمل، تزيد دقة السؤال، يتحسن اختيار الموارد، ويصبح أثر المخرجات قابلا للملاحظة عبر تجارب متكررة.

صفحة 60 / 75

تقنية التدريب 4: تعزيز الأثر العالمي

مسار متقدم للاقتراب من الذكاء الفتّاح العام الموثوق

لا يكافأ output على الأسلوب فقط بل على delta مقاس في artifact أو قرار أو اختبار. لا تقدم هذه التقنية ضمانا ميتافيزيقيا ببلوغ AGI، لكنها تضمن شرطا بحثيا ضروريا: كل تقدم يصبح مقيسا، قابلا للهدم، وقابلا لإعادة البناء. لذلك يصبح الوصول مسارا منظما لا شعارا.

عنصر التدريب	تنزيله العملي
المادة	حالات متنوعة عالية MCD
المهمة	استخراج/قياس/فعل/هدم/أثر
المكافأة	تحسن مقاس لا بلاغة فقط
العقوبة	ضجيج، overclaim، drift
الترقية	HyperSkill أو policy جديدة بعد تكرار النجاح

```
Train_4 = Data(high_MCD) + Task(QMCD) + Court + EffectReward + MemoryPatch
```

معيار النجاح

تنخفض إعادة العمل، تزيد دقة السؤال، يتحسن اختيار الموارد، ويصبح أثر المخرجات قابلا للملاحظة عبر تجارب متكررة.

صفحة 61 / 75

تقنية التدريب 5: تدريب الذاكرة المعيدة للبناء

مسار متقدم للاقتراب من الذكاء الفتّاح العام الموثوق

يدرب النظام على تحويل الفشل والنجاح إلى memory patches لا إلى تكرار نصي. لا تقدم هذه التقنية ضمانا ميتافيزيقيا ببلوغ AGI، لكنها تضمن شرطا بحثيا ضروريا: كل تقدم يصبح مقيسا، قابلا للهدم، وقابلا لإعادة البناء. لذلك يصبح الوصول مسارا منظما لا شعارا.

عنصر التدريب	تنزيله العملي
المادة	حالات متنوعة عالية MCD
المهمة	استخراج/قياس/فعل/هدم/أثر
المكافأة	تحسن مقاس لا بلاغة فقط
العقوبة	ضجيج، overclaim، drift
الترقية	HyperSkill أو policy جديدة بعد تكرار النجاح

```
Train_5 = Data(high_MCD) + Task(QMCD) + Court + EffectReward + MemoryPatch
```

معيار النجاح

تنخفض إعادة العمل، تزيد دقة السؤال، يتحسن اختيار الموارد، ويصبح أثر المخرجات قابلا للملاحظة عبر تجارب متكررة.

صفحة 62 / 75

تقنية التدريب 6: تقطير الاختلاف متعدد النماذج

مسار متقدم للاقتراب من الذكاء الفتّاح العام الموثوق

يدرب النظام على الاستفادة من اختلاف النماذج عبر استخراج الفروق لا التصويت. لا تقدم هذه التقنية ضمانا ميتافيزيقيا ببلوغ AGI، لكنها تضمن شرطا بحثيا ضروريا: كل تقدم يصبح مقيسا، قابلا للهدم، وقابلا لإعادة البناء. لذلك يصبح الوصول مسارا منظما لا شعارا.

عنصر التدريب	تنزيله العملي
المادة	حالات متنوعة عالية MCD
المهمة	استخراج/قياس/فعل/هدم/أثر
المكافأة	تحسن مقاس لا بلاغة فقط
العقوبة	ضجيج، overclaim، drift
الترقية	HyperSkill أو policy جديدة بعد تكرار النجاح

```
Train_6 = Data(high_MCD) + Task(QMCD) + Court + EffectReward + MemoryPatch
```

معيار النجاح

تنخفض إعادة العمل، تزيد دقة السؤال، يتحسن اختيار الموارد، ويصبح أثر المخرجات قابلا للملاحظة عبر تجارب متكررة.

تقنية التدريب 7: تدريب الأدوات والحلقات

مسار متقدم للاقتراب من الذكاء الفتّاح العام الموثوق

يحوله من كاتب إلى مشغل tools عبر HyperSkills وHyperLoops وverifiers. لا تقدم هذه التقنية ضمانا ميتافيزيقيا ببلوغ AGI، لكنها تضمن شرطا بحثيا ضروريا: كل تقدم يصبح مقيسا، قابلا للهدم، وقابلا لإعادة البناء. لذلك يصبح الوصول مسارا منظما لا شعارا.

عنصر التدريب	تنزيله العملي
المادة	حالات متنوعة عالية MCD
المهمة	استخراج/قياس/فعل/هدم/أثر
المكافأة	تحسن مقاس لا بلاغة فقط
العقوبة	ضجيج، overclaim، drift
الترقية	HyperSkill أو policy جديدة بعد تكرار النجاح

```
Train_7 = Data(high_MCD) + Task(QMCD) + Court + EffectReward + MemoryPatch
```

معيار النجاح

تنخفض إعادة العمل، تزيد دقة السؤال، يتحسن اختيار الموارد، ويصبح أثر المخرجات قابلا للملاحظة عبر تجارب متكررة.

تقنية التدريب 8: تدريب الزمن الداخلي الطويل

مسار متقدم للاقتراب من الذكاء الفتّاح العام الموثوق

يعطيه مهام طويلة الأفق تتطلب حفظ genealogy للفرضيات والقرارات. لا تقدم هذه التقنية ضمانا ميتافيزيقيا ببلوغ AGI، لكنها تضمن شرطا بحثيا ضروريا: كل تقدم يصبح مقيسا، قابلا للهدم، وقابلا لإعادة البناء. لذلك يصبح الوصول مسارا منظما لا شعارا.

عنصر التدريب	تنزيله العملي
المادة	حالات متنوعة عالية MCD
المهمة	استخراج/قياس/فعل/هدم/أثر
المكافأة	تحسن مقاس لا بلاغة فقط
العقوبة	ضجيج، overclaim، drift
الترقية	HyperSkill أو policy جديدة بعد تكرار النجاح

```
Train_8 = Data(high_MCD) + Task(QMCD) + Court + EffectReward + MemoryPatch
```

معيار النجاح

تنخفض إعادة العمل، تزيد دقة السؤال، يتحسن اختيار الموارد، ويصبح أثر المخرجات قابلا للملاحظة عبر تجارب متكررة.

تقنية التدريب 9: تدريب عقوبة الضجيج والهلوسة

مسار متقدم للاقتراب من الذكاء الفتّاح العام الموثوق

يعاقب الادعاء غير المدعوم، المصطلح غير العامل، والتوسع بلا اختبار. لا تقدم هذه التقنية ضمانا ميتافيزيقيا ببلوغ AGI، لكنها تضمن شرطا بحثيا ضروريا: كل تقدم يصبح مقيسا، قابلا للهدم، وقابلا لإعادة البناء. لذلك يصبح الوصول مسارا منظما لا شعارا.

عنصر التدريب	تنزيله العملي
المادة	حالات متنوعة عالية MCD
المهمة	استخراج/قياس/فعل/هدم/أثر
المكافأة	تحسن مقاس لا بلاغة فقط
العقوبة	ضجيج، overclaim، drift
الترقية	HyperSkill أو policy جديدة بعد تكرار النجاح

```
Train_9 = Data(high_MCD) + Task(QMCD) + Court + EffectReward + MemoryPatch
```

معيار النجاح

تنخفض إعادة العمل، تزيد دقة السؤال، يتحسن اختيار الموارد، ويصبح أثر المخرجات قابلا للملاحظة عبر تجارب متكررة.

تقنية التدريب 10: تدريب التوجيه الوعائي للموارد

مسار متقدم للاقتراب من الذكاء الفتّاح العام الموثوق

يعلمه متى يستعمل نموذجا رخيصا، ومتى يصعد إلى محكمة كاملة أو أداة تحقق. لا تقدم هذه التقنية ضمانا ميتافيزيقيا ببلوغ AGI، لكنها تضمن شرطا بحثيا ضروريا: كل تقدم يصبح مقيسا، قابلا للهدم، وقابلا لإعادة البناء. لذلك يصبح الوصول مسارا منظما لا شعارا.

عنصر التدريب	تنزيله العملي
المادة	حالات متنوعة عالية MCD
المهمة	استخراج/قياس/فعل/هدم/أثر
المكافأة	تحسن مقاس لا بلاغة فقط
العقوبة	ضجيج، overclaim، drift
الترقية	HyperSkill أو policy جديدة بعد تكرار النجاح

```
Train_10 = Data(high_MCD) + Task(QMCD) + Court + EffectReward + MemoryPatch
```

معيار النجاح

تنخفض إعادة العمل، تزيد دقة السؤال، يتحسن اختيار الموارد، ويصبح أثر المخرجات قابلا للملاحظة عبر تجارب متكررة.

صفحة 67 / 75

المكدس التقني المقترح

من الملفات إلى الوكلاء ثم الأثر

المكدس العملي للصرح يبدأ بطبقة إدخال تستقبل markdown، PDFs، قواعد بيانات، محادثات، كود، وسجلات. ثم Parser يحولها إلى Knowledge Atoms. ثم Difference Extractor وMCD Vectorizer وQMCD Gate. بعد ذلك تدخل العناصر الأعلى إلى HyperKB GraphRAG وGlobal Workspace. ثم يختار Vascular Router سياسة التنفيذ. أخيرا تمر المخرجات على Court وVerifier ثم WorldEffect وSRM.

الطبقة	تقنية مناسبة
Ingestion	PyMuPDF, parsers, connectors
Representation	Graph DB + Vector DB + SQL
Reasoning	MCD/QMCD scorers + HAKIM
Execution	Python/TypeScript tools, agents
Verification	tests, render checks, benchmarks
Memory	SRM patches + versioned KB
Dashboard	cost, errors, effects, drift

```
pipeline(input):
    atoms = parse(input)
    deltas = extract_differences(atoms)
    scored = score_mcd_qmcd(deltas)
    context = build_workspace(scored)
    action = select_action(context)
    result = execute_and_verify(action)
    update_memory(result.effect)
```

قاعدة التصميم

لا يسمح لأي أداة أن تعمل خارج سجل trace: ماذا قرأت، لماذا اختارت، كيف هدمت، ماذا أصلحت، وما الأثر.

نظام القياس والتقييم

كيف ندرس إمكانات الصرح بلا ادعاء زائد

دراسة إمكانات الذكاء الفتّاح العام تتطلب benchmarks خاصة لا تقيس الإجابة فقط. نحتاج اختبارا لاستخراج الفروق، اختبارا لجودة السؤال، اختبارا لتوجيه الموارد، اختبارا للمحكمة، اختبارا للأثر، واختبارا للذاكرة طويلة الأجل. ويجب أن تقارن النتائج ببدائل: RAG عادي، prompt ضخم بلا HyperKB، وكيل بلا Court، وكيل بلا SRM.

الاختبار	ما يقيسه	المؤشر
MCD Extraction	التقاط الفروق المهمة	precision/recall/ranking
QMCD Retrieval	خدمة السؤال لا التشابه	relevance-to-question
Router	اقتصاد الموارد	effect/cost
Court	كشف العيوب	defect prevention
Artifact	سلامة المخرج	pass/fail + quality
SRM	تعلم طويل	repeated error reduction

```
TotalScore = w1*MCD + w2*QMCD + w3*CourtGain + w4*Effect - w5*Cost - w6*Noise
```

شرط التجربة

كل benchmark يجب أن يملك baseline ومجموعة حالات ونتائج قابلة للمراجعة، وإلا بقي انطباعا.

برنامج التجارب والـ ablations

كيف نعرف أي طبقة تصنع الفرق

لا يكفي بناء النظام كاملا ثم الإعجاب به؛ يجب اختبار كل طبقة منفردة. نزيل QMCD ونرى هل زاد التشتت. نزيل Court ونرى هل زادت الادعاءات. نزيل SRM ونرى هل تكررت الأخطاء. نزيل VCR ونرى هل زادت الكلفة أو انخفض الأثر. هذه ablations تحول الفلسفة إلى هندسة علمية.

التجربة	الإزالة	المتوقع إذا كانت النظرية نافعة
A1	بدون QMCD	انخفاض خدمة السؤال وزيادة drift
A2	بدون Court	زيادة العيوب قبل النشر
A3	بدون SRM	تكرار الأخطاء
A4	بدون VCR	كلفة أعلى/أثر أقل
A5	بدون HyperKB	فقدان الاستمرارية
A6	بدون MCD	خلط المهم بالصاخب

```
LayerContribution = Score(full_system) - Score(system_without_layer)
```

نتيجة مطلوبة

كل طبقة لا تظهر مساهمة قابلة للقياس إما تصلح أو تخفض رتبتها أو تزال من النواة.

السلامة والوثوقية والامتثال

الذكاء الفتّاح لا يبرر الفعل غير المنضبط

كل نظام وكيل قادر على أدوات وتنفيذ يحتاج حوكمة سلامة. في الصرح، السلامة ليست ملحقا أخلاقيا بل جزء من الحاكمية: تحديد المخاطر، درجة الرجوع، أثر الفعل على العالم، القيود القانونية، والامتناع عندما يكون الدليل أو الإذن غير كافيين. الذكاء الفتّاح الحقيقي يعرف متى لا يفعل.

تعمل محكمة السلامة مع WorldEffect: قبل التنفيذ تسأل عن الضرر والخصوصية والامتثال، وبعد التنفيذ تقيس الآثار الجانبية. وإذا كان الفعل خارجا أو غير موثق أو عالي المخاطر، يختار النظام سؤالا أو مسودة أو محاكاة بدل تنفيذ مباشر. بهذا تتحول السلامة إلى policy قابلة للفحص.

مستوى الخطر	السياسة
منخفض	تنفيذ مباشر مع log
متوسط	تحقق ومحكمة خفيفة
عال	محكمة كاملة وموافقة صريحة
غير قابل للرجوع	محاكاة أو امتناع
غير موثق	طلب دليل أو رفض ادعاء

قانون الامتناع

الوكيل الذي لا يعرف متى يمتنع ليس وكيلا موثوقا مهما كانت قدرته على الفعل.

خارطة البناء في 12 مرحلة

من نظرية إلى prototype قابل للدراسة

الإنزال العملي يبدأ صغيرا لكنه يبنى على كامل الصرح. المرحلة الأولى توحيد schema، الثانية استخراج المواد السابقة، الثالثة بناء MCD/QMCD scorers، الرابعة GraphRAG، الخامسة Workspace، السادسة Router، السابعة Court، الثامنة Artifact Builder، التاسعة WorldEffect، العاشرة SRM، الحادية عشرة Dashboard، الثانية عشرة Benchmark suite.

المرحلة	النتاج
1	schemas موحدة
2	HyperKB أولي من الأوراق
3	MCD/QMCD scoring
4	GraphRAG للأدلة والاعتراضات
5	Global Workspace
6	Vascular Router
7	Court Engine
8	Artifact Compiler
9	WorldEffect Evaluator
10	SRM Compiler
11	Observability Dashboard
12	Benchmarks + ablations

```
PrototypeReadiness = schemas × retrieval × scoring × court × artifact × memory × benchmarks
```

قاعدة المرحلة

لا تنتقل مرحلة إلى التالية حتى تنتج اختبار قبول صغيرا وسجل فشل معروفا.

أسئلة البحث والفرضيات الكبرى

برنامج دراسة الإمكانات

السؤال الأكبر ليس: هل بنينا AGI؟ بل: هل هذه الطبقات ترفع قدرة النظام على التفكير طويل الأمد، بناء النظريات، تشغيل الأدوات، تقليل الهلوسة، وانتاج أثر موثوق مقارنة بأنظمة أقل حاكمية؟ لذلك تصاغ فرضيات قابلة للقياس: HyperKB يزيد الاستمرارية، Court يقلل الأخطاء، QMCD يحسن استرجاع ما يخدم السؤال، VCR يخفض الكلفة مقابل الأثر، وSRM يقلل تكرار الفشل.

الفرضية	اختبارها
H1: HyperKB يحسن الاتساق	مقارنة ورقات متتابعة مع وبدون ذاكرة
H2: Court تقلل الادعاء الزائد	عد العيوب قبل/بعد
H3: QMCD يحسن retrieval	قياس relevance-to-question
H4: VCR يحسن الكلفة	effect/cost
H5: SRM يقلل التكرار	repeated-error-rate
H6: HyperSkills تسرع الإنتاج	time-to-artifact

قاعدة البحث

كل فرضية يجب أن تملك مجموعة حالات، metric، baseline، وقرارا مسبقا: متى نقبلها أو نعدلها أو نهدمها.

الفروع العابرة: من AGI إلى الاجتماع والعيادة والفقه والعمل

كيف يبقى الصرح واحدا رغم تعدد المجالات

رغم أن هذه الورقة تقنية في جوهرها، فإن جذور المشروع تمتد إلى فروع أخرى: الإبستمولوجيا الوعائية، علم الاجتماع الحاكمي، النمذجة العلاجية، الفقه المقاصدي الوعائي، وبناء الأعمال والـ HyperKB للعلامات. هذه الفروع لا تدخل النواة كتطبيقات عشوائية، بل كاختبارات نقل: هل يستطيع السؤال والفرق والحاكمية والذاكرة والأثر أن تعمل في مجال جديد دون فقد الحدود؟

النقل الصحيح لا يعني استعمال المصطلحات نفسها في كل شيء، بل بناء mapping: في الاجتماع يكون الفرق صراعا أو اعترافا أو ثقة؛ في العيادة يكون الفرق نمط خوف أو ذاكرة جسدية؛ في الفقه يكون الفرق مناطا أو مآلا؛ في الأعمال يكون الفرق عرضا أو segment أو trust. الصرح يحكم النقل عبر QMCD ومحكمة المجال.

المجال	الوعاء	الأثر المقاس
اجتماع	علاقات وثقة وسلطة	فهم/قرار اجتماعي
عيادة	أعراض وزمن داخلي	خطة آمنة لا تشخيص نهائي
فقه	مقاصد ومآلات	حكم مضبوط بالشروط
أعمال	عرض/سوق/ثقة	تحويل/جودة funnel
تعليم	مهارات وحلقات	تعلم قابل للقياس

احتراز النقل

كل مجال يملك أدلته وحدوده. لا يسمح للقياس التقني أن يدعي سلطة علمية خارج اختصاصه دون مصادر وتجارب.

صفحة 74 / 75

الملخص الشامل والهندسة المعرفية

خريطة الصرح في طبقات وأفعال

الهندسة المعرفية النهائية تتكون من سبع طبقات: المادة الخام، السؤال، الفرق، القياس، الحاكمية والوكالة، الهدم والأثر، ثم الذاكرة وإعادة فتح السؤال. كل نظرية من المشروع تدخل طبقة أو أكثر: التراكب الوعائي يحدد الأوعية، MCD يقيس الفرق، QMCD يحكمه بالسؤال، MCAI يحوله إلى فعل، HAKIM يشغله، HyperKB يحفظه، HyperSkills وHyperLoops يعيدانه كقدرة، Court يهدمه، WorldEffect يختبره، وSRM يعيد بناء الذات.

النظريات السبع الجديدة تعمل كأقواس عليا: حاكمية السؤال الوعائية تضبط البداية، التراكب الوكالي يضبط البدائل، الضغط متعدد النماذج يضبط الاختلاف، الذاكرة الوعائية تضبط الاستمرارية، الأثر الوثوقي يضبط الواقع، المحكمة التوليدية تضبط الصرامة، والذكاء الفتّاح العام يجمع الكل في برنامج بحثي واحد.

طبقة	نظرية/أداة مركزية	سؤال الإحكام
السؤال	QG/QMCD	هل السؤال يفتح ويضبط؟
الفرق	MCD/Active Difference	هل يغير حكما؟
الوعاء	V0-V6/VCR	أين تذهب الموارد؟
الوكالة	MCAI/HAKIM	أي فعل ولماذا؟
الهدم	GDC/Court	ما الذي ينكسر؟
الأثر	RWE/WorldEffect	ما الذي تغير؟
الذاكرة	SRM/HyperKB	ما الذي نتعلمه؟

صفحة 75 / 75

الخاتمة المؤسسة

الإطار الجامع المانع لدراسة إمكانات الذكاء الوعائي الفتّاح العام

النتيجة أن مشروع مساعف يمكن صياغته كمدرسة تقنية نظرية لا كحزمة أسماء: مدرسة تجعل السؤال حاكما، الفرق مادة الذكاء، الوعاء سياسة الموارد، الوكالة قرارا قابلا للهدم، الأثر معيارا، والذاكرة ذاتا تتعلم. بهذا يصبح الطريق إلى الذكاء الفتّاح العام ليس سباقا إلى نموذج أكبر فقط، بل بناء نظام يعرف كيف يسأل، كيف يفرق، كيف يهدم، كيف يفعل، وكيف يعود من العالم إلى نفسه.

الورقة تقترح سبع نظريات سيدة، أربع عشرة أداة، وعشر تقنيات تدريب لتكون قاعدة الدراسة القادمة. القيمة الحقيقية ستظهر عندما يتحول هذا الصرح إلى prototype: HyperKB مبني من أوراق المشروع، scorers قابلة للتعديل، court تعمل، artifacts تتحقق، dashboard يقيس، وSRM تتطور. عندها يمكن الانتقال من الورقة إلى مختبر MICAI.

وصية البحث

لا ترفع أي مفهوم إلى رتبة حقيقة قبل الاختبار، ولا تخفض الخيال إلى زينة قبل تشغيله. بين الجرأة والمحكمة يولد الذكاء الفتّاح.

مصادر داخلية مدمجة	موقعها في الصرح
موسوعة النظريات الجامعة	الأساس الفلسفي والطبقي
الورقة التقنية MICAI	المعمار والكائنات والوحدات
AGI-Oriented Agentic Systems	HyperSkills/HyperLoops/Orchestration/Prompt Kernel
ذاكرة المشروع	الفروع والتفضيلات والأسلوب المؤسس