SOLO R&D / ALYANNA VELANDO

Epistemic-Centric AI Design

Knowability, the Mangravecola experiment
and the Epistemic Trace Layer



Alyanna Velando

Independent research / A4 reading edition

archive.alyannavelando.com/epistemic-centric-ai/

SOLO R&D / RESEARCH OVERVIEW

Epistemic-Centric AI Design

How could an interface help people inspect the evidence, inferences and assumptions supporting an AI response before they act on it?

01 · Trace

Identify the support

Distinguish sources, user context, inference and assumptions.

02 · Graph

Inspect dependencies

See which claims rely on which premises.

03 · Orient

Guide verification

Locate gaps between apparent authority and visible support.

The problem: apparent knowledge

Fluent AI output can appear better grounded than it is. This research connects questions of knowability and representation to how people interpret that output. The Mangravecola thought experiment explores how a concept can remain usable even after its supposed real-world referent is withdrawn.

The proposed Epistemic Trace Layer

The interaction model makes the status and dependencies of claims available for inspection. It proposes a structured account of what supports an output, rather than a display of the model’s raw internal chain of thought.

Support human judgement

A glanceable cue could lead into detailed inspection and targeted verification. The proposed Mirage Indicator concerns the mismatch between apparent authority and visible support; it is not a probability that an answer is false.

STATUS OF THE WORK

Independent conceptual R&D. The interaction model and Mirage Indicator are proposals; their implementation, measurement and effect on human judgement require validation.

Read online → archive.alyannavelando.com/epistemic-centric-ai/

Contents

Knowability and the Finite Knower 6

1. The Problem . . . 6
2. The Finite Knower . . . 6
3. Knowing as a Pre-dictive Structure . . . 7
4. Pre-dictability as an Operational Route into Knowability . . . 8
5. Memory, Awareness and the Temporality of Knowing . . . 10
6. Language as Epistemic Infrastructure . . . 11
7. The Mangravecola Experiment . . . 12
8. From Fiction to Epistemic Loss . . . 13
9. Knowability as a Relation Rather Than a Property . . . 14
10. Recoverability . . . 15
11. Returning to the Original Properties . . . 15
12. Working Proposition . . . 16
13. The Mangravecola Boundary . . . 17

Knowability: The AI Loop 18

10.1 Before the Word . . . 18
10.2 Concept Construction Through Context . . . 19
10.3 Referential Collapse . . . 19
10.4 Representation Is Enough for Use . . . 20
10.5 The Reset Problem . . . 20
10.6 Did the Model Ever “Know” the Mangravecola? . . . 21
10.7 The Mirage Reappears . . . 22
10.8 The Recursive Structure . . . 23
10.9 A Possible AI Formulation . . . 23

10.10 The Hard Question 24

Epistemic Trace Layer **25**

1. Origin 25
2. Core Proposition 25
3. Epistemic Categories 26
4. From Source Provenance to Epistemic Provenance 26
5. Output as an Inspectable Structure 27
6. Dependency Structure 27
7. The Mirage Problem 28
8. The HMI Shift 28
9. Why the Mangravecola Matters 29
10. Hallucinations as an Interaction Problem 29
11. Working Product Principle 29
12. Working Name 30
13. Core Line 30

Epistemic Trace Layer: Interaction & Architecture **31**

14. From Output-Centric AI to Epistemic-State-Centric AI 31
15. A Shift in Epistemic Responsibility 32
16. Sector Consequences 32
16.1 Law 33
16.2 Medicine 33
16.3 Finance 34
16.4 Engineering and Safety-Critical Systems 34
16.5 Defence / C2 / Operator Systems 35
16.6 Science and Research 35
17. Hallucination as an Interaction Problem 36

18. Epistemic Mirage → Mangravecola → Trace Layer 36
19. The Mirage Indicator 37
20. Mirage Is Not Uncertainty 38
21. Local Mirage 39
22. Two-Level Interaction 39
Level 1 - Glanceable 39
Level 2 - Inspectable 39
23. Three Product Layers 40
1. Trace 40
2. Graph 40
3. Score / Indicator 40
24. Core Product Architecture 40
25. Status Quo vs Proposed Model 41
26. The Research Line 41
27. Core Design Principles 43

PART 01 / RESEARCH NOTES

Knowability and the Finite Knower

A working reconstruction of prediction, representation, language, memory and the Mangravecola problem

1. The Problem

A question about knowability appears simple until the word knowable itself is examined.

Suppose a knower is finite.

It has limited memory, limited processing capacity, limited time, limited access to information, and limited means of representation.

A first hypothesis might therefore be:

Certain structural properties of a domain determine whether that domain can be known by a finite knower.

Candidate properties could include:

- compressibility
- stability
- bounded dependency
- finite description length

At first glance, these appear to be properties of the domain.

But the formulation already hides several unresolved questions:

What is a finite knower? What counts as knowing? Is knowability a property of the domain itself, or of the relation between a domain and a knower? Does a domain remain knowable if the means required to access it disappear?

Before evaluating the candidate properties, the object under investigation has to be made more precise.

2. The Finite Knower

The phrase finite knower is underspecified.

A human is finite. An artificial system is finite.

A hypothetical epistemic agent with bounded memory and bounded computation is finite.

But these are not finite in the same way.

For a human, relevant boundaries may include:

$$K_h = \{\text{memory, attention, lifespan, perception, language, awareness}\}$$

For an artificial system:

$$K_a = \{\text{compute, architecture, context, representation, training, access}\}$$

The relevant question is therefore not simply:

Is the domain finite enough to be known?

but:

Finite relative to what kind of knower?

This suggests an initial relational formulation:

$$\text{Knowability} \neq f(D)$$

where D is the domain alone.

A more plausible starting point is:

$$\text{Knowability} = f(D, K)$$

where:

- D = domain
- K = knower

The same domain may therefore differ in accessibility depending on the capacities of the agent attempting to know it.

3. Knowing as a Pre-dictive Structure

One way to make knowing less abstract is to examine what acts associated with knowledge actually do.

Consider:

- knowing
- thinking
- inferring
- guessing
- recognising
- becoming aware

These differ in epistemic strength.

Yet they appear to share a structural feature:

they constrain expectation.

To know something is not merely to contain information.

Knowing produces some commitment concerning what is, what should follow, what should be found, or what should remain consistent under examination.

This can be described as a pre-dictive structure.

Here pre-diction should not be reduced to forecasting future events.

It can instead be understood more broadly as the articulated production of an expectation prior to subsequent confirmation.

For example:

I know that Paris is in France.

This knowledge generates expectations.

If I consult a reliable map, I expect Paris to appear in France.

If someone places Paris in Germany, I expect the statement to be false.

If asked where Paris is located, I expect my answer to converge on France.

The knowledge therefore constrains the possible outcomes compatible with it.

In this sense:

$$\text{Knowledge} \rightarrow \text{Expectation}$$

or more cautiously:

$$K_n \Rightarrow E$$

where K_n represents a knowledge-state and E an expectation licensed by that state.

Different epistemic acts may generate expectations with different levels of warrant:

$$\text{Guessing} \rightarrow E_{\text{weak}}$$
$$\text{Inference} \rightarrow E_{\text{premise-constrained}}$$
$$\text{Knowing} \rightarrow E_{\text{stronger}}$$

The distinction between them is therefore not necessarily the presence or absence of pre-dictive structure.

It may instead concern the basis, reliability and warrant of the expectation being produced.

4. Pre-dictability as an Operational Route into Knowability

This produces an alternative route into the original problem.

Instead of beginning with:

What makes something knowable?

we can temporarily ask:

What allows a finite knower to generate reliable expectations about a domain enough to pre-dict something about it?

This reframing gives the candidate structural properties a common function.

Stability

If the relevant behaviour of a domain changes without sufficient regularity, previous states cease to constrain future or unobserved states.

$$\text{Stability} \downarrow \Rightarrow \text{Predictive reliability} \downarrow$$

Bounded Dependency

If determining any state requires resolving an indefinitely expanding network of dependencies, a finite agent may never be able to close the inference.

$$\text{Dependencies} \to \infty$$

while:

$$\text{Resources}(K) < \infty$$

produces an accessibility problem.

Compressibility

A finite knower cannot retain every possible observation separately.

Regularities allow many observations to be represented by fewer structures.

$$n \text{ observations} \to m \text{ representations}$$

where:

$$m \ll n$$

Compression therefore permits finite resources to support expectations over a larger space than could be memorised case by case.

Finite Description Length

If the representation necessary to capture the relevant structure of a domain requires an unbounded description, a strictly finite knower cannot fully instantiate that representation.

These four properties may therefore share a deeper role:

they help make inference and expectation possible under finite epistemic resources.

This does not yet prove that they are necessary conditions of knowability.

But it gives a reason why they might matter.

The stronger hypothesis becomes:

> Finite Knowability ⇒ Finite Predictive Structure

or perhaps:

> $\text{Knowability}_K(D) \propto \text{Ability}_K(\text{generate reliable expectations about } D)$

The exact relation remains open.

5. Memory, Awareness and the Temporality of Knowing

Finiteness is not only computational.

It also appears through memory and awareness.

Suppose a person knows x at time t_1.

At t_2, the person has completely forgotten x.

Several different questions now appear:

Did they know x at t_1? Do they know x at t_2? Does the information still exist somewhere in the system?

Is knowing dependent on present accessibility? Does awareness of knowing constitute a different epistemic state?

This makes time relevant.

A knowledge-state can be represented as:

> $K(x, t)$

rather than simply:

> $K(x)$

So:

> $K(x, t_1) = 1$

does not necessarily imply:

> $K(x, t_2) = 1$

The knower may have changed while the proposition has not.

This suggests that knowability itself may need a temporal parameter.

> $\text{Knowability}(D, K, t)$

rather than:

```
Knowability(D, K)
```

This becomes especially important once knowledge can be lost.

6. Language as Epistemic Infrastructure

The knower does not normally interact with a domain without mediation.

Humans use representations.

Among the most powerful of these is language.

A word can act as a stable handle for a larger conceptual structure.

It can:

- compress distinctions;
- preserve categories;
- permit recall;
- enable recombination;
- support abstraction;
- allow transfer between minds.

A concept that would otherwise require repeated reconstruction can be indexed by a compact symbol.

Schematically:

```
Complex Conceptual Structure → (encoding) Word
```

and later:

```
Word → (retrieval) Conceptual Structure
```

This makes language relevant to the problem of finite knowing.

It acts as a form of cognitive compression.

A finite knower does not need to rebuild an entire conceptual network every time it wishes to use it.

The representation carries part of the load which complicates the original formulation again.

Knowability may not depend only on:

```
D × K
```

but also on the representational system through which the knower accesses the domain:

```
D × K × R
```

where:

- D = domain
- K = knower
- R = representation

The question changes from:

What properties must the domain possess?

to:

What properties must the relationship between domain, representation and knower possess?

7. The Mangravecola Experiment

Consider the word:

Mangravecola

t0 - No Concept

Before the term is introduced, the listener possesses no concept identified as Mangravecola.

$$M_{concept} = 0$$

The string has no established referent or conceptual content for the listener.

t1 - Referential Construction

The following description is supplied:

“ A Mangravecola is the plant owned by my cousin.”

The listener now possesses a conceptual object.

$$M_{concept} = 1$$

and believes that it possesses a physical referent:

$$M_{referent} = 1$$

The word has become a stable handle.

It can now support additional operations:

Where is the Mangravecola? Who owns it? Does it need water? What kind of plant is it?

t2 - Referential Collapse

The description is then withdrawn:

“ There is no cousin. There is no plant. Mangravecola was invented.”

The physical referent disappears:

$$M_{referent} = 0$$

But the conceptual object does not.

$$M_{concept} = 1$$

The listener can still:

- remember the Mangravecola;
- describe it;
- deny its physical existence;
- recognise the word;
- discuss the deception through which it was introduced.

Therefore:

$$Existence_{referent} \neq Existence_{concept}$$

The destruction of the referent does not retroactively destroy the conceptual event.

8. From Fiction to Epistemic Loss

Now introduce another transformation to the Mangravecola Experiment.

Suppose that:

- every person involved dies;
- every written record disappears;
- every digital record is destroyed;
- every system retaining the term is reset;
- no surviving agent possesses evidence from which Mangravecola could be reconstructed.

At t_3:

$$Access(M) = 0$$

For future knowers, Mangravecola may become epistemically indistinguishable from something that was never conceived.

Yet the historical event remains different.

Compare:

World A

Mangravecola was never conceived

World B

Mangravecola was conceived → known → all traces lost

At t_3, future observers may have no method capable of distinguishing World A from World B.

Nevertheless:

$$A \neq B$$

The epistemic states available to future agents are identical.

The histories are not. This produces the Mangravecola Problem:

Can something cease to be knowable without ceasing to have been true?

And more specifically:

Can a domain contain facts that are no longer recoverable by any finite knower even though those facts genuinely occurred?

If yes, then:

$$\text{Truth} \nRightarrow \text{Present Knowability}$$

and:

$$\text{Past Knowability} \nRightarrow \text{Future Knowability}$$

9. Knowability as a Relation Rather Than a Property

The thought experiment suggests that knowability may not be a permanent intrinsic property of an object or domain.

A more complete formulation might be:

$$\text{Knowability} = f(D,K,R,T)$$

where:

D = Domain
K = Knower
R = Representation/epistemicaccessstructure
T = Time

This allows the same domain to move through different epistemic states without changing its historical truth.

For example:

$$\text{Knowable}(D, K, R, t_1) = 1$$

while:

$$\text{Knowable}(D, K, R, t_2) = 0$$

if the representation or evidence required for access disappears.

The property therefore belongs not simply to D, but to a temporary epistemic relation.

10. Recoverability

This introduces another concept that may be useful:

recoverability

Something may be presently unknown while remaining recoverable.

For example:

$$\text{Unknown}_{\text{now}}$$

but:

$$\text{Evidence} > 0$$

therefore:

$$\text{Recoverability} > 0$$

By contrast, the final Mangravecola state has:

$$\text{Evidence} = 0$$
$$\text{Memory} = 0$$
$$\text{Representation} = 0$$

and therefore potentially:

$$\text{Recoverability} = 0$$

This suggests three different states that should not be collapsed:

Known
Unknown but recoverable
Epistemically irrecoverable

The last of these is especially important.

An irrecoverable fact is not necessarily a false fact.

It may simply be a fact for which no surviving path to knowledge remains.

11. Returning to the Original Properties

The original four properties can now be reconsidered inside this larger structure.

Perhaps:

compressibility, stability, bounded dependency and finite description length are not properties of knowability itself.

They may instead be properties that help maintain:

> Recoverability

and:

> Reliable inference

within a finite epistemic system.

This yields a layered model:

```
Structural Properties(D)
↓
Representability(D, R)
↓
Accessibility(K, R)
↓
Inference/Prediction
↓
Knowing
```

The four proposed properties may act near the beginning of this chain.

Language, memory and representation act further downstream.

Loss can occur at several points.

A domain may remain structurally regular while becoming unknowable because its evidence disappeared.

Conversely, a domain may be richly observable but remain difficult to know because a finite knower lacks a representational system capable of compressing its structure.

12. Working Proposition

The resulting proposition is therefore not:

Knowability is predictability.

Nor:

language creates reality.

Nor:

finite domains are knowable.

It is closer to:

For a finite knower, knowing requires some structure capable of constraining expectation. The capacity to form and preserve that structure depends not only on the domain, but on the knower, its representations, and the temporal survival of epistemic access.

Formally:

```
Knowability(D,K,R,T)
```

with reliable epistemic access requiring some relation between:

```
Structure(D)
Capacity(K)
Representation(R)
Persistence(T)
```

The open question is then no longer merely:

What makes a domain knowable?

It becomes:

Under what structural and representational conditions can a finite knower construct, retain and recover enough of a domain to generate warranted expectations about it?

13. The Mangravecola Boundary

The Mangravecola produces a useful boundary case because its lifecycle can be represented explicitly:

```
t0: ¬Concept
t1: Concept + Believed Referent
t2: Concept + ¬Referent
t3: Historical Concept + ¬Accessible Representation
```

The transition from t_2 to t_3 is epistemically more interesting than the fiction itself.

It separates:

```
Did exist as a concept
```

from:

```
Can now be known to have existed
```

If every path connecting those propositions is destroyed, their truth values remain different even though no surviving observer can establish the difference.

That is the central tension:

Reality can preserve a distinction after knowledge has lost the means to recover it. And if that is possible, knowability cannot simply be treated as an eternal property of whatever is true.

PART 02 / RESEARCH NOTES

Knowability: The AI Loop

The Mangravecola experiment becomes more interesting when the finite knower is no longer human.

An LLM is also finite.

It has bounded context, bounded compute, bounded access, bounded persistence, and a specific representational architecture.

But unlike a human knower, its apparent knowledge can be strongly dependent on whether a representation is currently available inside the active context.

This makes the earlier formulation directly applicable:

```
Knowability = f(D, K, R, T)
```

For an AI system:

```
D = Domain
K = Model
R = Available Representation
T = Contextual/Temporal Access
```

The Mangravecola provides a useful test because the concept was created inside the interaction itself.

10.1 Before the Word

Before the term Mangravecola was introduced, the model had no established concept associated with it.

$$M_{concept} = 0$$

The token sequence could still be processed linguistically.

The model could attempt analogy, morphology or probabilistic association.

But it had no grounded representation of Mangravecola as a specific concept.

It could only infer.

This is important.

The ability to produce a plausible interpretation is not the same as already possessing the concept.

10.2 Concept Construction Through Context

A description is then introduced:

“A Mangravecola is the plant owned by my cousin.”

The system can now use the word operationally.

It can:

- refer to the plant;
- answer questions about it;
- preserve conversational consistency;
- infer consequences from the description;
- distinguish it from other concepts.

Behaviourally, the state now resembles:

$$M_{concept} = 1$$

But what changed?

Not the external world.

Only the representation available to the model changed.

The concept became usable because a local structure had been supplied inside context.

This suggests:

$$\text{Operational Concept Use} \neq \text{External Grounding}$$

A model can manipulate a concept coherently even when the concept has no real-world referent.

10.3 Referential Collapse

The description is then withdrawn:

There is no cousin. There is no plant. The Mangravecola was invented.

The system updates.

The physical referent becomes:

$$M_{referent} = 0$$

while the concept remains available:

$$M_{concept} = 1$$

The model can now reason about the Mangravecola as a fictional or invented object.

It can understand:

“The Mangravecola never physically existed.”

The representation survives the collapse of the referent.

This means that coherent concept-use cannot by itself establish correspondence with reality.

The model may preserve:

```
Conceptual Consistency
```

while lacking:

```
Referential Truth
```

This is one of the central tensions of AI epistemology.

10.4 Representation Is Enough for Use

The Mangravecola reveals something simple but important.

A model does not need a real referent in order to generate a functional representation.

It only needs enough structure to constrain subsequent output.

In simplified form:

```
Input → Representation → Prediction → Output
```

Once a representation exists, it can constrain later predictions.

That may look like knowledge.

But several very different states can produce similar behaviour:

```
Grounded Knowledge
Contextual Representation
Fabricated but Internally Consistent Representation
```

All three may allow fluent continuation.

Therefore:

```
Fluency + Consistency + Inference ⇏ Grounding
```

The system may behave as though it knows because the representation is operationally sufficient.

That does not yet tell us what kind of knowing is occurring.

10.5 The Reset Problem

Now remove the context.

Suppose the model no longer has access to the conversation in which Mangravecola was defined.

At t_1:

$$K(M, t_1) = 1$$

in the weak operational sense that the system can use the concept correctly within the active interaction.

At t_2, after context loss:

$$K(M, t_2) = 0$$

or more precisely:

$$Access(M, t_2) = 0$$

The concept may no longer be recoverable by that system instance.

Nothing about the original conceptual event has changed.

What changed is access.

This creates the same distinction that appeared earlier with human memory:

$$Past\ Accessibility \nRightarrow Present\ Accessibility$$

For an LLM, however, this transition may happen not through biological forgetting, but through:

- context truncation;
- session reset;
- retrieval failure;
- memory boundaries;
- missing external state;
- model replacement;
- absence from training data.

The epistemic loss is architectural rather than biological.

But structurally, the problem is similar.

10.6 Did the Model Ever “Know” the Mangravecola?

This produces a difficult question.

If a model can:

- use the term correctly;

- preserve its properties;
- draw valid inferences from it;
- distinguish it from related concepts;
- revise the representation when contradicted;

then in what sense did it not know the Mangravecola?

But if the representation:

- existed only locally;
- had no external grounding;
- disappeared once context disappeared;

then in what sense did it know it?

The problem may therefore be badly framed if reduced to:

“Did the model know it: yes or no?”

A better formulation may be:

What kind of epistemic access did the model have, and under what representational conditions did that access remain available?

This suggests that AI “knowing” may need to be decomposed into states such as:

```
Representation
Operational Access
Inference
Persistence
Grounding
Recoverability
```

rather than treated as a single binary property.

10.7 The Mirage Reappears

The experiment also exposes a problem already visible in LLM interaction.

A model can produce a response that is:

- fluent;
- coherent;
- well structured;
- inferentially sophisticated;

while still operating on a false or misaligned representation.

The output can therefore create the appearance of understanding before grounding has been established.

The Mangravecola makes this especially clear because the false referent was deliberately introduced.

The model behaved coherently around it.

Nothing in the fluency of the response exposed the fiction.

This produces a useful distinction:

```
Performance ≠ Epistemic Status
```

The quality of the output does not uniquely identify the quality of the underlying relation to truth.

A system may reason well inside a false frame.

That is not the same failure as random hallucination.

It may be more subtle. The representation can be internally coherent and inferentially productive while being externally wrong. Potentially increasing the risk of “mirage judgement”.

10.8 The Recursive Structure

The Mangravecola experiment is unusual because the problem emerged through the system it later describes.

The sequence was:

```
Human disagreement with LLM
↓
Invented word
↓
LLM constructs local representation
↓
False referent is removed
↓
Concept remains operational
↓
Contextual persistence becomes visible
↓
The resulting thought experiment
↓
is applied back to LLM knowing
```

The system became part of the experiment.

It was not merely asked to analyse an abstract theory of knowledge.

Its own behaviour helped expose the distinction.

10.9 A Possible AI Formulation

The earlier relational model can therefore be specialised for AI systems:

```
AI Knowability = f(D,M,R,C,T)
```

where:

```
D = Domain
M = Model
R = Representation
C = Context/Retrieval Conditions
T = Temporal Persistence
```

This does not define machine knowledge.

It defines some of the conditions under which something can remain epistemically available to the system.

A more cautious formulation would therefore be:

An AI system can only operate on what is representationally available to it, and the ability to use a representation does not by itself establish whether that representation is grounded, true, persistent or recoverable.

This moves the question away from:

“Does the AI know?”

toward:

What is the relation between the system, the representation it is using, the domain that representation refers to, and the conditions under which that relation survives?

10.10 The Hard Question

The Mangravecola leaves one unresolved question:

If a system can reason coherently about a concept, revise it, infer from it and use it correctly within context, but has no independent grounding and loses the concept when the representation disappears, what exactly should we call that state?

Knowledge?

Access?

Temporary representation?

Operational understanding?

Prediction over a locally constructed structure?

The answer matters because modern AI systems can increasingly produce behaviours associated with knowing without giving us a simple way to determine what kind of epistemic state produced them.

The Mangravecola does not solve that problem but it sure makes the boundary harder to ignore.

PART 03 / RESEARCH NOTES

Epistemic Trace Layer

From Knowability to Inspectable AI

1. Origin

The concept did not begin as a UI idea.

It emerged from a sequence of questions around:

finite knowers knowability prediction language representation the Mangravecola thought experiment LLM grounding Epistemic Mirage AI hallucinations Human-Machine Interaction

That progression eventually exposed a design problem:

LLM interfaces expose semantic output, but hide the epistemic structure that supports it.

The result is that epistemically different statements are often rendered with the same apparent authority.

A retrieved fact, an inference, an assumption and a speculation may all appear simply as:

sentence sentence sentence sentence

The user sees the answer.

They do not see the epistemic status of the answer.

That is the gap.

2. Core Proposition

Current conversational AI interaction is usually represented as:

```
Prompt → Output
```

The proposed model introduces an inspectable intermediate layer:

```
Prompt → Epistemic Structure → Output
```

The purpose is not to expose raw chain-of-thought.

It is to expose a structured, user-facing account of what kind of epistemic operation supports each part of the response.

The interface should not show:

“everything the model thought.”

It should show:

“what kind of epistemic support this output depends on.”

3. Epistemic Categories

A first working taxonomy:

SOURCE

Information grounded in a retrieved or externally verifiable source.

USER INPUT

Information supplied by the user and currently treated as part of the active context.

MODEL PRIOR

Information generated from the model’s existing learned representation without current external retrieval.

INFERENCE

A conclusion derived from available premises.

ASSUMPTION

A premise being treated as true without independent verification.

SPECULATION

A plausible possibility proposed without sufficient evidence to establish it.

CORRECTION / UPDATE

A revision to a previously held representation after new information is introduced.

UNRESOLVED

A dependency, claim or ambiguity that remains open.

These categories can coexist within the same paragraph or conclusion.

4. From Source Provenance to Epistemic Provenance

Existing AI interfaces may already expose forms of document provenance:

this claim came from this source.

The proposed layer extends this into:

why this claim exists in the answer at all.

For example:

“Smith v Jones established X.”

could expose:

SOURCE-GROUNDED

- retrieved legal database
- source verified
- passage available while another apparently identical sentence might expose:

MODEL-GENERATED CLAIM

- no retrieved source
- citation unverified
- requires verification

The distinction is not cosmetic.

It changes what the user knows about the claim.

5. Output as an Inspectable Structure

The answer remains readable as normal text.

But paragraphs, claims or transitions become inspectable.

A user may click a passage and open a compact side panel showing:

Epistemic status: INFERENCE Depends on: User statement A + Source B Assumptions: 1 External grounding: partial Unresolved dependency: C

The inverse interaction should also work.

Clicking:

ASSUMPTION could highlight every part of the answer that depends on an unverified premise.

Clicking:

SOURCE could highlight only source-grounded claims.

Clicking:

SPECULATION could expose where the system has moved beyond established evidence.

This keeps the primary answer visually clean while making its epistemic structure accessible on demand.

6. Dependency Structure

Reasoning does not always follow:

```
A → B → C → D
```

A more realistic structure may be:

```
A + B → C
B + E → F
C + F → G
```

The interface can therefore expose a compact dependency graph.

Selecting G could show:

depends on A depends on B depends on E contains inference C contains unresolved dependency F

This makes the answer inspectable as a network of epistemic dependencies, rather than merely as prose.

7. The Mirage Problem

The Epistemic Mirage emerges partly because different epistemic states are flattened into the same visual presentation.

The user sees:

```
Retrieved Fact
Inference
Assumption
Speculation
```

but the interface renders all four as ordinary prose.

Thus:

```
Semantic Output
```

is visible, while:

```
Epistemic Structure
```

is hidden.

The proposed layer adds that second dimension.

8. The HMI Shift

The traditional user instruction is:

Verify the AI.

But this places the full epistemic burden on the user.

A better interaction principle may be:

Do not ask the user to verify everything. Show them what requires verification.

This changes the role of the interface.

The UI is no longer only a container for generated language.

It becomes a system for epistemic orientation.

9. Why the Mangravecola Matters

The Mangravecola thought experiment exposed the difference between:

```
Operational Concept Use
```

and:

```
External Grounding
```

An LLM can use a representation coherently even when the referent is false or nonexistent.

Therefore:

```
Fluency + Consistency + Inference ⇏ Grounding
```

The epistemic trace layer responds directly to that problem.

Instead of asking the user to infer grounding from language quality, it exposes the status of the representation supporting the output.

10. Hallucinations as an Interaction Problem

Hallucination is usually treated as a model failure.

But user harm depends on more than whether the model generated a false claim.

A more complete interaction model is:

```
Hallucination + Human Trust + Low Epistemic Visibility → Risk
```

This means hallucination severity is partly an HMI property.

A false claim that is visibly unverified behaves differently from the same claim presented with the same visual authority as a verified fact.

Therefore the design problem is not only:

reduce hallucinations.

It is also:

reduce the probability that epistemically weak content is mistaken for epistemically strong content.

11. Working Product Principle

```
Semantic Output + Epistemic Structure
```

The system should make both available.

The user should be able to inspect:

- what was sourced;
- what came from them;
- what was inferred;
- what was assumed;
- what remains speculative;
- what was corrected;
- what depends on what.

The objective is not transparency for its own sake.

It is better-calibrated human judgement.

12. Working Name

For now:

Epistemic Trace Layer

Alternative:

Epistemic Provenance Interface

The first name emphasises the product layer.

The second emphasises epistemic provenance.

13. Core Line

The central design principle is:

The interface should not only show what the model says. It should expose the epistemic status of what it says.

A complementary principle is:

Do not ask the user to verify everything. Show them what requires verification.

PART 04 / RESEARCH NOTES

Epistemic Trace Layer: Interaction & Architecture

Extended Product / HMI Consequences

14. From Output-Centric AI to Epistemic-State-Centric AI

Most conversational AI interfaces still treat the generated answer as the primary object.

The basic interaction is:

```
Prompt → Output
```

The user receives a response and must independently determine:

- what is verified;
- what is inferred;
- what is speculative;
- what comes from the user;
- what comes from retrieved evidence;
- what may have been fabricated.

The epistemic burden therefore sits largely with the human.

A traceable epistemic interface changes the interaction into:

```
AI Output → Visible Epistemic Structure → Targeted Human Verification
```

The shift is therefore:

```
Output Centric → Epistemic State Centric
```

The answer is no longer treated as a finished object.

It becomes an inspectable epistemic object.

The user does not only see:

what the AI says.

They can also inspect:

what the claim depends on, how it was formed, what kind of claim it is, what has been verified, and where uncertainty remains.

This changes trust from a global judgement-

“Do I trust the AI?”

-to a local one:

“I trust this passage because it is source-grounded.” “I should verify this one because it depends on an inference.” “This section is explicitly speculative.”

That is a fundamentally different trust model.

15. A Shift in Epistemic Responsibility

The current status quo often looks like:

```
AI produces
↓
Human interprets
↓
Human verifies
↓
Human discovers error
```

The interface itself contributes very little to helping the user determine where verification is necessary.

An epistemic trace layer redistributes that work:

```
AI produces
↓
Interface exposes epistemic status
↓
Human evaluates selectively
↓
Verification occurs before action
```

The design principle becomes:

Do not ask the user to verify everything. Show them what requires verification.

This does not remove human responsibility.

It makes that responsibility more actionable.

The interface stops acting merely as a text-delivery surface and begins functioning as an epistemic orientation system.

16. Sector Consequences

The same architecture produces very different value depending on domain.

16.1 Law

Legal AI has an obvious failure mode:

a fabricated case citation and a real one may look typographically identical.

Without epistemic provenance:

Smith v Jones [2024] UKSC 15 established that... appears authoritative whether it is real or invented.

With an epistemic layer, the same claim could expose:

SOURCE-GROUNDED

- retrieved from verified legal database
- source available
- passage linked or:

MODEL-GENERATED

- no retrieved authority
- citation not verified
- verification required

The status quo changes from:

> Verification after reading

to:

> Verification guided by epistemic status

The professional no longer has to treat every statement as equally suspicious, nor every fluent statement as equally authoritative.

16.2 Medicine

A medical recommendation could expose whether it comes from:

- clinical guideline;
- patient record;
- retrieved study;
- model inference;
- generalisation;
- unsupported assumption.

For example:

“Consider X intervention.”

could be tagged as:

GUIDELINE-GROUNDED or:

INFERENCE FROM PATIENT CONTEXT or:

GENERAL MODEL PRIOR - NOT PATIENT-SPECIFIC

The interface would therefore support not only what the recommendation is, but what kind of reasoning relation supports it.

This could turn conversational AI from a text generator into a second layer of clinical reasoning support.

16.3 Finance

A statement such as:

“This company appears undervalued.”

could be decomposed into:

```
Verified Market Data
Sector Assumptions
Margin Forecast
Speculative Scenario
```

Instead of receiving one authoritative sentence, the user sees the epistemic mixture inside it.

The distinction between:

fact and:

thesis becomes visible.

16.4 Engineering and Safety-Critical Systems

Here the dependency graph becomes especially important.

Suppose a recommendation depends on:

```
A + B → C
C + D → E
E + F → Recommendation
```

If B and F are unverified assumptions, the recommendation may be much more fragile than its final phrasing suggests.

An epistemic interface can expose:

dependency depth unresolved nodes source coverage assumption concentration fragile branches

The question changes from:

“What does the AI recommend?”

to:

“How structurally fragile is the path supporting this recommendation?”

16.5 Defence / C2 / Operator Systems

In operator-facing environments, the problem becomes:

> Correctness + Decision Latency + Trust Calibration

An operator may not have time to inspect a long answer.

The interface therefore needs a glanceable summary:

SOURCE-BACKED INFERRED ASSUMPTION-DEPENDENT UNRESOLVED

A compact epistemic layer could help answer:

What can I act on immediately? What needs verification? What is only an AI-generated possibility? Which recommendation rests on a fragile dependency?

This is particularly relevant where the cost of over-trust and under-trust are both high.

16.6 Science and Research

LLM-generated scientific synthesis often visually flattens:

- retrieved evidence;
- synthesis;
- hypothesis;
- extrapolation;
- unsupported leap.

An epistemic interface could make the literature-review process inspectable.

Instead of:

“The literature suggests X.”

the user could see:

RETRIEVED EVIDENCE → Papers A, B, C

SYNTHESIS → Model-generated aggregation

EXTRAPOLATION → Not directly supported by cited papers

This would make AI-assisted research less dependent on the user detecting epistemic shifts purely from prose.

The sector examples all follow the same underlying shift: AI answers become inspectable epistemic objects rather than undifferentiated generated text.

17. Hallucination as an Interaction Problem

Hallucination is often framed as:

```
Hallucination = Model Failure
```

But user harm does not follow automatically from the hallucination itself.

A more complete chain is:

```
Hallucination + Human Trust + Low Epistemic Visibility → Risk
```

Therefore:

hallucination severity is partly an HMI property.

The same false claim behaves differently depending on presentation.

Case A

“This may be incorrect. I could not verify the citation.”

The claim is weakly grounded and visibly weakly grounded.

Case B

“Smith v Jones [2024] UKSC 15 established...”

No provenance. No warning. Authoritative language.

The underlying factual error may be identical.

The interaction risk is not.

Therefore AI safety has at least two layers:

```
Model Reliability
Interaction Transparency
```

The goal is not only:

reduce hallucinations.

It is also:

reduce the probability that epistemically weak content is interpreted as epistemically strong content.

That is where the Epistemic Mirage becomes an HMI problem rather than only an AI-literacy problem.

18. Epistemic Mirage → Mangravecola → Trace Layer

The emerging line can now be represented as:

```
Epistemic Mirage
↓
Human mistakes fluency for epistemic competence
↓
Mangravecola
↓
Operational concept use without referential grounding
↓
Epistemic Trace Layer
↓
Make epistemic status inspectable
```

The relationship can therefore be expressed as:

Epistemic Mirage = diagnosis

Mangravecola = mechanism

Epistemic Trace Layer = intervention

This is no longer a single feature idea.

It is the beginning of a coherent HMI research direction.

19. The Mirage Indicator

Once epistemic categories are made explicit, they can potentially be summarised.

This introduces a new layer:

```
Epistemic Trace → Mirage Indicator
```

The purpose of the indicator is not to estimate whether the answer is true.

It must not be interpreted as:

“Mirage 78% = 78% probability the answer is false.”

That would create false precision.

Instead, the Mirage Indicator would represent something closer to:

the degree to which the apparent epistemic strength of an output exceeds the visible support behind it.

Conceptually:

```
Mirage Risk = f(Unsupported Claims, Inference Depth, Assumption Load, Speculation, Source Coverage, Unre...
```

This formula is truncated in the source PDF.

with:

```
Mirage Risk ↓
```

as:

```
Verified Grounding ↑
```

A more abstract expression is:

```
Mirage ∝ (Perceived Epistemic Authority) / (Visible Epistemic Support)
```

This is a conceptual design relation, not yet a validated metric.

20. Mirage Is Not Uncertainty

This distinction is critical.

A speculative response is not necessarily high-Mirage.

For example:

“One possible explanation is X.”

The statement is speculative, but its epistemic status is correctly communicated.

Therefore:

```
Speculation High
```

does not necessarily imply:

```
Mirage High
```

By contrast:

“X is the cause.”

with:

- no retrieved evidence;
- hidden assumptions;
- speculative dependencies;
- high-certainty presentation;

may produce a much stronger Mirage.

The Mirage should therefore measure:

misalignment between epistemic status and epistemic presentation rather than simply:

amount of uncertainty.

This is essential.

Otherwise the system would punish legitimate exploration, hypothesis generation and creative reasoning.

21. Local Mirage

The Mirage need not exist only at answer level.

It can be calculated or represented locally.

For example:

Paragraph 1 - Mirage: Low Paragraph 2 - Mirage: High Paragraph 3 - Mirage: Medium

Clicking paragraph 2 might expose:

no retrieved source high inference depth assumption-dependent user-context only unresolved dependency

The user therefore does not receive the instruction:

“verify the whole answer.”

They receive:

“verify here.”

This is the difference between generic caution and actionable epistemic guidance.

22. Two-Level Interaction

The system could therefore support two levels.

Level 1 - Glanceable

A compact state:

Mirage: LOW or:

Mirage: MEDIUM or:

Mirage: HIGH

This helps non-expert users.

Level 2 - Inspectable

Clicking reveals:

- epistemic categories;
- provenance;
- dependency graph;
- unresolved assumptions;
- verification status;
- source coverage.

Thus:

```
Glanceability + Inspectability
```

coexist.

The user does not need to understand epistemology in order to benefit from epistemically structured interaction.

23. Three Product Layers

At this point the concept contains three distinct product layers:

1. Trace

What epistemic category supports this claim?

```
Claim → Category
```

2. Graph

What does this claim depend on?

```
Dependencies → Claim
```

3. Score / Indicator

How large is the mismatch between apparent authority and visible support?

```
Trace + Graph → Mirage Indicator
```

These correspond roughly to:

explainability inspectability glanceability

The concept therefore stops being a single UI component.

It becomes a system of epistemic interaction design.

24. Core Product Architecture

The full architecture can now be written as:

```
Prompt
↓
Model Processing
↓
Epistemic Classification
↓
Dependency Mapping
↓
Semantic Output
Epistemic Trace
↓
Mirage Indicator
↓
Human Inspection
↓
Targeted Verification
↓
Decision/Action
```

This makes the epistemic state part of the interaction itself.

25. Status Quo vs Proposed Model

Current

```
Prompt → Answer → User Trust Judgement
```

The human must infer epistemic quality primarily from:

- prose quality;
- citations;
- tone;
- prior trust;
- personal expertise.

Proposed

```
Prompt → Answer + Epistemic Structure → Calibrated Human Judgement
```

The interface contributes explicit information about epistemic status.

That is the actual status-quo shift.

26. The Research Line

The work now forms a coherent progression:

Epistemic Mirage

Why humans misread AI output

↓

Mangravecola

How operational representation can exist without referential grounding

↓

Epistemic Trace Layer

How interfaces could expose epistemic status

↓

Dependency Graph

How users could inspect the support structure behind claims

↓

Mirage Indicator

How epistemic risk could be compressed into an actionable cue

↓

Domain Adaptation

How the model changes for law, medicine, finance, science, engineering and operator systems

This is a cumulative research progression.

Not:

“one article led to another random idea.”

But:

one epistemic problem progressively generated a mechanism, an interaction model and a possible product architecture.

27. Core Design Principles

The guiding principles are:

The interface should not only show what the model says. It should expose the epistemic status of what it says.

and:

Do not ask the user to verify everything. Show them what requires verification.

A further principle is:

The goal is not to make AI appear less intelligent. It is to make the basis of its apparent intelligence inspectable.