Where GPT Behavior Comes From

Four Philosophers, a linguistic stack, and a practical map of deployed systems

Michael Stoyanovich

Disclaimer

This paper is intended for informational and educational purposes only. The views and analyses presented - particularly those related to ethics, policy, and AI system design - reflect the author's interpretations and do not constitute legal, regulatory, or professional advice. Readers are encouraged to critically assess the content and consult appropriate experts or authorities before applying any concepts discussed herein. The author assumes no liability for any decisions or actions taken on the basis of this work.

Abstract[1]

When a GPT hallucinates, where in the stack did things go wrong? LLM commentary often collapses into a binary: either the system "understands," or it is "only" next-token prediction. That framing hides where behavior actually comes from in deployed GPT systems. This essay offers a pragmatic map that links a four-philosophers lens (Wittgenstein, Lewis, Dennett, and Nagel) to a linguistics stack, and then maps both onto a plain-text GPT system.[2] The map separates the system into three interdependent lanes: Control (the instruction layer and how a request is packaged), Core (the model, which predicts what text is likely to come next given what it can see), and Outside-core (the surrounding system components: tokenization, decoding, context handling, policy enforcement, and optional tools or retrieval). The point is practical, not metaphysical: to attribute strengths and failure modes to the right place, and to put controls where they actually improve reliability and auditability. Speech acts—asserting, requesting, committing, refusing, hedging—are the bridge between philosophy, linguistics, and system design. The mapping is aimed at policy, product, and governance readers who need a clearer frame for evaluation and accountability. A running example is used throughout: a compliance workflow that summarizes guidance with citations and a freshness check.

Guide for policy and governance readers

This essay is a conceptual map intended to support evaluation and governance decisions. This analysis assumes a text-only interface; multimodal systems largely expand Outside-core with additional I/O pipelines and grounding mechanisms.

The argument does not require machine learning (ML) fluency; technical terms are optional detail.

---

[1]Author's note: The author is not a professional linguist, academic philosopher, or machine learning (ML) researcher. This essay offers a pragmatic synthesis developed for interpretive clarity in policy and governance contexts, drawing on secondary and primary sources where possible. The goal is usability and intellectual hygiene, not disciplinary novelty. This is a practitioner-oriented synthesis, not a new theory of meaning or mind.

[2]This essay assumes a text-only interface; multimodal deployments are briefly noted in Appendix D.

- Start with Sections 3, 5, and 6 (lanes, speech acts, and implications).
- Use the compressed table (end of section 4) as a checklist: identify the linguistic issue, then locate whether it is Control, Core, or Outside-core.
- When a term slows the read, jump to Appendix B (plain-language notes), Appendix C (glossary), or Appendix D (linguistics stack), then return.

Figure 1. *The three lanes of a deployed GPT system*

Control	Core	Outside-core
• Instructions and packaging • Roles (system/developer/user) • Templates, constraints, formatting	• The model itself • Predicts what text comes next • Learns patterns from training data	• Everything around the model • Tokenization, decoding, context handling • Policy enforcement, tools/retrieval, logging

1. Why "Does it understand?" is misleading

The question "does the model really understand?" is not meaningless; it is simply overworked. In practice it serves as a rhetorical lever: inflating systems into agents ("as if" they had humanlike agency) or deflating them into stochastic party tricks. Neither posture helps when a concrete task must be executed, evaluated, governed, and—crucially—trusted in context.

Start with the system, not the slogan. A deployed GPT is an end-to-end system: (i) instructions that frame the task and set priorities (Control), (ii) the model that predicts likely next text from the context it can see (Core), and (iii) the deployment layer that turns that into a usable product (Outside-core). The rest of this essay treats these as three interdependent lanes. Linguistics provides a matching stack on the language side—discourse, pragmatics, speech acts, syntax, lexicon, and more. The four philosophers then function as guardrails: they help keep claims about meaning, commitment, competence, and experience in bounds. This is not a claim about what GPTs are in any ultimate sense; it is a practical frame for locating where behavior comes from in deployed systems, and where policy and controls can actually act.

2. Four lenses that limit over-claiming

The framework used here is not deployed as a set of theses to be defended, but as a set of constraints on what can responsibly be inferred from GPT behavior.

Wittgenstein anchors meaning in use and rule-governed practice. On this view, the relevant question is not whether a string of text "contains" meaning, but whether the output functions correctly within a language-game: a task with norms, roles, stakes, and standards of success.

Lewis emphasizes convention, coordination, and the management of common ground—what is treated as settled, what is presupposed, what is at issue, what has been committed to, and

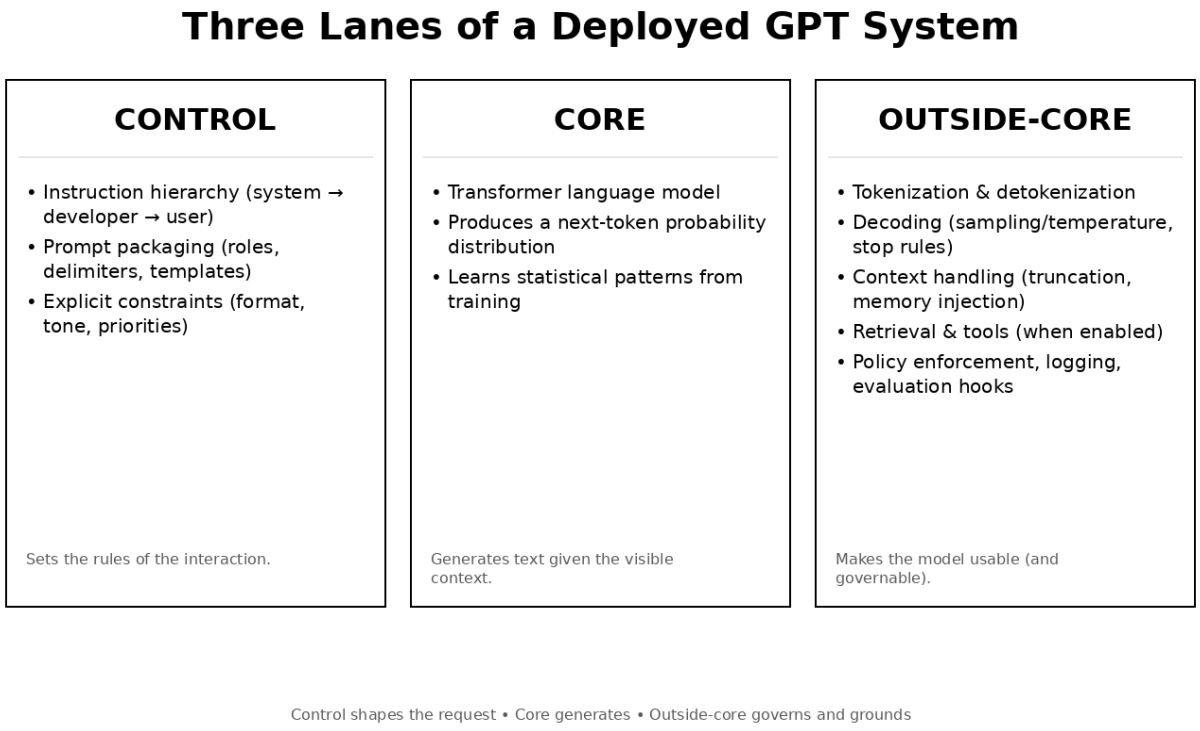
what can be revised. This lens foregrounds conversational bookkeeping: the ongoing update of commitments that helps keep human–GPT discourse coherent.

Dennett offers a pragmatic discipline for interpreting complex behavior. The intentional stance can be useful as a predictive shortcut—talking as if the system "believes" or "wants" something often improves short-term forecasting of outputs—by tracking what the human interlocutor expects the GPT to do next. But the stance becomes a trap when it is treated as an ontological promotion—that is, when a predictive convenience is mistaken for a claim about the system's real nature. Competence does not entail comprehension; a fluent explanation by the GPT does not mean it has stable understanding of what it is saying.

Nagel provides the boundary condition. A system can generate first-person-like discourse without that implying subjective experience—without anything it is like to be that system. More generally: fluent talk about the world is not itself a tether to the world. The lens is less an accusation than a reminder: language can simulate presence without providing it. There is no "there, there" that the system itself gives access to. Nothing in the system warrants the claim that there is anything it is like to be a GPT.

These lenses reappear in the discussion of speech acts, evaluation, and governance, where the mapping is applied as an interpretive and operational discipline.

Figure 2. *Three lane visual of a deployed GPT system.*



3. Three lanes: Control, Core, Outside-core

A common mistake is to talk about "the model" as if it were the whole product. In practice, what people use is a deployed GPT system built from three interdependent lanes:

1. Control: the instruction layer—system/developer/user messages, templates, delimiters, and formatting rules. Control sets the game: what is being asked, what constraints apply, and what counts as an acceptable answer.

2. Core: the model itself. Although it is trained to predict text, the behavior that falls out of that objective can include surprisingly capable patterns. Given the text it can see, it predicts what text is most likely to come next. This is where most language competence comes from (grammar, phrasing, patterns of explanation), but it is not a built-in fact checker.

3. Outside-core: the deployment layer around the model—tokenization, decoding settings, context-window handling (what is kept vs dropped), policy enforcement, and any retrieval/tools. In production, this layer often decides whether answers are safe, structured, sourced, and consistent.

This separation matters because many behaviors that look "cognitive" are actually properties of the surrounding system components (e.g. stale retrieval): decoding settings that change confidence and verbosity; context truncation that creates contradictions; and retrieval/tools that change what information is available at answer time.

Some deployments blur the boundaries. Retrieval or tool calls can be triggered automatically, and some products add extra inference-time steps. Even then, the lanes remain a useful diagnostic: is a behavior coming from the model, from the instructions, or from the deployment layer?

A further caveat: "Core" is used here as a governance abstraction, not as a claim that modern model internals are architecturally simple. Contemporary systems may include attention variants, routing mechanisms, multimodal components, optimized kernels, and distributed inference paths. Those details matter for engineering, performance, and some forms of auditability, but the diagnostic question remains the same: is the behavior best attributed to instructions, learned model behavior, or surrounding runtime/deployment machinery?

4. Mapping linguistics onto GPT systems

This section overlays two checklists—the linguistics stack and the three lanes—so a reader can locate where a problem is likely introduced. With the system lanes in view, the next step is to lay a linguistics stack over them: to see where a plain-text GPT has a natural mapping, and where it does not.

A plain-text GPT system does not map to all layers of linguistics. Phonetics and phonology have no appropriate mapping in a purely textual system. Likewise, psycholinguistics and neurolinguistics do not map literally: the substrate is not a human cognitive process, nor a brain.

But many layers do map pragmatically:

- **Orthography/graphology** maps to outside-core I/O and learned conventions in the core, with control shaping formatting norms.

- **Morphology** maps weakly via subword tokenization; token boundaries can

resemble morphemes, but the match is incidental and imperfect.

- **Lexicon, syntax, and compositional semantics** map primarily to core behavior, where distributed representations support structure-sensitive generation. Control and outside-core constraints can shape the form of that generation without changing the underlying learned structure.

- **Discourse** maps to attention over a finite context window, while outside-core context management imposes hard limits on continuity and coherence.

- **Pragmatics** spans all three lanes: control supplies intent cues and priorities; the core reproduces pragmatic patterns unevenly; outside-core policy enforcement can override stance.

Two distinctions clarify "reference," which is often treated as a single problem:

Deixis/indexicals (here, now, this, that) can often be stabilized by spelling out the context in text—what "this" refers to, what "here" means, and which time "now" is anchored to. For example, instead of "fix this," write "fix the second paragraph under 'Governance'," or instead of "what's happening now?" write "as of December 2025, what is the current status of X?" When it fails, the reason is usually simple: the cue is missing, ambiguous, or has been dropped from the context the system can see. The system then fills the gap with a plausible referent, which can be wrong even when the prose sounds confident.

Empirical verification—being tethered to what is actually true—is not intrinsic to the core model. A GPT can produce a confident answer that reads like a policy memo and still be wrong, outdated, or quietly fabricated. A useful rule of thumb: if the output must be correct, not merely coherent, verification cannot be optional. When verification is required, it typically has to be provided outside-core: retrieval from approved repositories, citations tied to specific passages, freshness checks for time-sensitive topics, and human review when the consequences are real (benefits eligibility, compliance posture, legal interpretation).

In table terms: deixis is often fixable in Control and Outside-core (by naming exactly what "this/that/here/now" points to). Empirical verification lives almost entirely in Outside-core —because it requires a pipeline of sources and checks rather than language fluency.

In the running example, "cite sources" is a Control rule, while the choice of what to retrieve (and what to ignore) sits in Outside-core; the model's fluency can hide that mismatch.

A recurrent failure shows up at the seams between lanes. Control may demand "cite sources" or "don't guess," while Outside-core retrieval returns material that is partial, conflicting, or stale. The core can then produce a confident answer that stitches the pieces together anyway. For example, a prompt asks for the latest policy position, but retrieval returns an older memo; unless the Outside-core layer enforces a freshness check, the system may blend the older memo into a "current" answer. The issue is less "bad reasoning" than a mismatch between the speech act being demanded (a grounded assertion) and the evidence actually available.

Mitigations are mostly architectural rather than rhetorical: bind claims to retrieved passages; check freshness and authority; require uncertainty marking when evidence is thin; and fall

back to clarification or refusal when the system cannot meet the requested act.

What follows is the condensed diagnostic checklist version of that overlay.

How to read the mapping in the table below

Columns are system lanes (Control, Core, Outside-core); rows are linguistic layers; tags index the four philosophers.

Use the table as a checklist when diagnosing behavior: locate the linguistic phenomenon, then ask which lane is responsible. Where there is no appropriate plain-text GPT mapping, the layer is preserved and marked "NO MAPPING." Plain-language technical notes appear in Appendix B; a glossary is provided in Appendix C.

Tags: W = Wittgenstein, L = Lewis, D = Dennett, N = Nagel (limit).

The table is not meant to be memorized; it is a diagnostic checklist.

Running example (policy)

A compliance team asks an internal GPT assistant: "Summarize the new guidance and tell us what to change in our policy. Cite sources and flag uncertainty." The system replies confidently, but it leans on an outdated secondary source and misses a key caveat in the primary guidance.

Compressed table view (key rows):

- Walkthrough (one row):
  - Pick a row—for example, Deixis/indexicals (this/that/here/now). Read across.
- Control: did the prompt pin the reference ("this" = which document/section/date)?
- Core: can the model resolve it from what it can see, or does it guess a referent?
- Outside-core: did the system inject missing context (file name, time, tool output), or did truncation drop it?
- Diagnostic move: if the answer points to the wrong "this," treat it as a reference failure, not "bad reasoning."
- Governance move: require explicit anchors ("this = …") in high-stakes workflows, and log what context was provided.

Linguistics layer (tags)	CONTROL (message hierarchy)	CORE (transformer)	OUTSIDE-core (runtime + I/O)
Deixis / Indexicals [W,L]	pin "here/now/this/that" via explicit context ("In this doc…", "In 2025…")	resolves indexicals via textual cues in-window; brittle when cues are missing	context injection (time, locale, tool outputs) can stabilize reference
Empirical verification / World-tethering [N(LIMIT),W,L,D]	require sources, quoting, and "don't guess" rules	no inherent truth-check; plausible continuations absent grounding	tools/RAG/browsing (if present) provide external checks; otherwise none
Discourse / Text	delimiting/quoting/	attention over context	context mgmt.:

linguistics [L]	scope defines "context"	window supports coherence (within window)	truncation, memory injection, summarization; KV-cache/runtime constraints
Pragmatics (speaker meaning) [W,L]	intent cues, priority rules, constraints	pragmatics-ish patterns (inconsistent)	policy/safety layers can override stance; tool gating
Speech acts (assert/request/commit/refuse/hedge) [W,L]	explicit act-shaping: "summarize", "argue", "refuse if…", "ask clarifying questions"	act-shaped continuations (asserting, hedging, complying) without truth guarantee	policy enforcement/refusals; constrained output channels (JSON/schema); tool calls as "acts"
Syntax (structure) [D]	formatting can make structure easier/harder (lists, code blocks, etc.)	structure-sensitive behavior via attention/position (emergent)	constrained decoding can enforce structure even if core would drift
Phonology / Phonetics [—]	NO MAPPING (plain text)	NO MAPPING (plain text)	NO MAPPING (plain text)

5. Speech acts as the operational hinge

If a single linguistic concept deserves pride of place in an LLM interpretation map, it is speech acts: asserting, requesting, committing, refusing, hedging.

Speech acts capture what an utterance *does* in interaction. That matters because GPT use is already speech act structured—often explicitly so. Instructions such as "summarize," "answer," "refuse if unsafe," "ask clarifying questions," "cite sources," and "output JSON" are not decorative; they are act-shaping constraints. They specify what kind of move the system is being asked (or allowed) to make.

Speech acts are used here as interaction categories—not as a claim that the system has intentions. They describe what an utterance does in a workflow (assert, refuse, ask), which is exactly what governance can constrain and audit.

In the running example, "Summarize…" is a request; "cite sources" and "flag uncertainty" constrain what counts as an acceptable assertion.

This is where the four lenses converge cleanly:

- **Wittgenstein**: a response is a move in a language-game; correctness is measured by fit to the practice.
- **Lewis**: moves update commitments; coherence is scorekeeping across turns.
- **Dennett**: act-shaped fluency tempts intentional attribution; the stance can predict behavior, but should not be mistaken for a warrant of understanding.

- **Nagel**: speech-act performance does not entail subjective experience; it is functional participation, not lived perspective.

The pragmatic payoff is immediate: reliability improves when the system is constrained to make the right kinds of moves—particularly when assertions are bound to sources, uncertainty marking is required, and inquiry is mandated when context is missing.

6. Implications for evaluation, reliability, and governance

The point of the framework is not to win an argument about what GPTs "really are," but to change what teams build and how they govern it. With that in view, the mapping distills into a small set of moves that materially improve reliability, auditability, and civil trust.

For product teams: governance moves that actually change outcomes

- **Bind high-stakes assertions to evidence.** Require that factual claims be tied to retrieved passages (or approved repositories), not just "sounds right" fluency.
- **Enforce reference anchoring ("this/that/here/now").** In high-stakes workflows, require explicit anchors ("this = …") and log what context was provided.
- **Treat freshness and authority as first-class controls.** Add freshness checks for time-sensitive topics and prefer authoritative sources; when evidence is stale or thin, force clarification, uncertainty marking, or refusal.
- **Log the instruction stack.** Preserve the system/developer/user instruction layers and active policies that shaped the act being performed.
- **Log what the system could see.** Snapshot the context window inputs (including truncation, summaries, and memory inserts) so outputs can be audited and reproduced.
- **Govern memory like a system of record.** Define what is summarized vs retained, who/what can write to memory, and how memory is versioned and reviewed after changes.

Many production failures occur at the seams between Control and Outside-core: Control demands a grounded act ("cite sources," "don't guess," "latest as of today"), but Outside-core retrieval returns evidence that is stale, partial, or conflicting, and the core model smooths the mismatch into fluent prose as if it were resolved.

The rest of this section unpacks why these moves follow from the four lenses and the three lanes.

This philosophical mapping points to a simple discipline: place critique and control where they belong.

In the simplest terms: evaluation is mostly a Wittgenstein/Lewis question (meaning-in-use and common ground). Reliability often depends on Outside-core (the deployment layer). Governance is act-shaping plus auditability: what rules were applied, what evidence was

used, and what the system could see at the time.

Evaluation should not be reduced to fluency. Lewis-style questions are especially useful: does the system keep the conversational "score" consistent across turns—what is treated as settled, what is merely assumed, and what the system has committed itself to? Does it preserve those commitments when asked again later, and does it revise them cleanly when conditions change (e.g., "given X, not Y")? Or does it answer each turn as if it were a fresh start, creating subtle contradictions? Wittgenstein-style questions matter too: does the output fit the game—its role, norms, stakes, and definition of success?

Reliability often lives in Outside-core. If the system loses context, it may contradict prior commitments. If the output is forced into a schema, it may look rigorous while remaining untrue. If retrieval or tools are added, answers can be tied to sources—but now the system can fail in new ways: stale retrieval, incorrect citations, and brittle tool calls. Once evaluation and reliability are located, governance can be stated plainly: set rules for which acts the system may perform (assert, recommend, refuse, escalate), under what evidentiary conditions, and how each act is logged and auditable.

Governance as act-shaping under audit

Governance becomes more legible when framed as act-shaping under audit. The practical question is not whether the model "understands," but under what conditions it is allowed to assert, when it must ask, what it must cite, and how its outputs can be checked and traced. In the running example, this means two simple requirements: (1) the answer must be tied to the retrieved passages, and (2) the system must log which passages were used so the claim can be checked later.

A Lewisian framing makes the audit target explicit: the evolving common ground—what the system treated as given, what it added, and what it revised. In a GPT system, that "common ground" is implemented as a moving context window plus whatever memory, retrieval, and tool outputs are injected around it.

Auditability is therefore not only about recording the final output. It is about preserving enough context to explain why an output was locally reasonable inside the system: the instruction stack, the context the system could see, the evidence it retrieved, and the constraints that shaped the response.

Minimum audit artifacts typically include (in plain terms):

- Instruction stack snapshots (what rules and instructions were in force for this request: system/developer/user messages, active policies, and guardrails).
- Context-window snapshots (what the system could "see" when it answered, including any truncation and any summaries or memory inserts that replaced earlier text).
- Retrieved or tool-produced evidence (what was fetched or generated outside the model, when it was fetched, and exactly what passages were tied to citations).
- Decoding and constraint configuration (what output formats were forced, what stop conditions applied, and what sampling settings affected variability).
- Output plus declared act constraints (what the system was required to do: cite sources, mark uncertainty, ask clarifying questions first, refuse under specific conditions, and so on).

For systems that maintain "memory" or summarized context, a minimum governance

discipline includes:

- What gets summarized vs retained (and when).
- Who or what is allowed to write to memory (and under what conditions).
- How memory is versioned and reviewed, especially after policy or model updates.

7. Conclusion

A plain-text GPT system is neither a mind nor a simple text engine. It is an end-to-end system that produces language-like behavior through the interaction of Control, Core, and Outside-core. Linguistics offers a useful breakdown of what language does; the three-lane view offers a useful breakdown of where behavior comes from in production. The four-philosophers lens keeps interpretation disciplined: meaning as use (Wittgenstein), commitments and common ground (Lewis), competence without assuming comprehension (Dennett), and limits on claims about experience and world-contact (Nagel). Speech acts make the framework operational. They explain why the system can sound socially competent, why anthropomorphic readings are tempting, and where governance controls can be placed to improve reliability without over claiming what the system is.

Ethics, Disclosure, and Acknowledgements

Ethical Considerations

This paper does not draw on private, sensitive, or personally identifiable data. All examples are hypothetical, anonymized, or derived from public sources. No formal human-subjects research was conducted, and no institutional ethics review was required. All citations seek to conform to academic standards.

The broader ethical implications of the arguments developed herein concern public misinterpretation, policy design, and stakeholder responsibility in AI deployment. These implications are intended to provoke critical discussion and inform future regulatory and design frameworks.

Use of AI Tools

AI language models - most notably OpenAI's ChatGPT - were used during the writing process as interlocutors: for brainstorming, structuring sections, and testing rhetorical clarity. These tools were instrumental in refining transitions, surfacing edge cases, and challenging internal consistency.

This meta-use aligns with the paper's themes. Interacting with generative AI during authorship provided firsthand insight into the very limitations this paper analyzes: fluency without grounding, responsiveness without perspective, and the ease with which stylistic coherence can be mistaken for conceptual depth.

Responsibility for all ideas, arguments, and conclusions lies solely with the human author.

Acknowledgments

The author wishes to thank informal readers who provided critical feedback on earlier drafts. Their questions, challenges, and encouragement materially improved the final manuscript. Special thanks to those who questioned assumptions, pushed for clearer synthesis, and reminded the author that philosophy and engineering are not separate disciplines - they are simply perspectives on design.

No institutional support, funding, or affiliation contributed to this work. All errors and omissions are the author's alone.

Disclosure Statement

This work was conducted independently, without institutional affiliation, funding, or external influence. The views expressed are the author's alone and do not represent any current or former employer. No financial or professional conflicts of interest are declared.

License & Attribution

To cite this paper:

Stoyanovich, Michael. *Where GPT Behavior Comes From: Four philosophers, a linguistics stack, and a practical map of deployed systems.* Version 1.3.2 (June 2026).

https://www.mstoyanovich.com

Version History and Document Status

This is a living document. As generative AI systems and their use evolve, this paper will be periodically updated to incorporate new empirical findings, theoretical insights, and policy developments. Major revisions are recorded here to preserve transparency and scholarly traceability.

Version	Date	Description
V1.3.2	June 2026	Added a brief clarification that "Core" is used as a governance abstraction rather than a claim that modern model internals are architecturally simple; noted that contemporary systems may include attention variants, routing, multimodal components, optimized kernels, and distributed inference paths while preserving the Control / Core / Outside-core diagnostic frame.
V1.3.1	December 2025	Added a "For product teams" governance callout at the start of Section 6 to surface practical implications (evidence-binding, anchoring deixis, freshness/authority checks, instruction/context logging, memory governance); added a short bridge sentence framing these moves as the applied payoff of the four lenses and three lanes.
V1.3.0	December 2025	Moved the condensed table and its on-ramp to the end of Section 4; tightened Section 4 flow; refined title/subtitle to "Where GPT Behavior Comes From."
V1.2.9	December 2025	Added table "on-ramp" improvements: worked example walkthrough, explicit Section 4 framing, and "not meant to be memorized" guidance to reduce cognitive load.
V1.2.8	December 2025	Incorporated critique-driven "misread defenses": clarified stance (not ontology), added emergence nuance, clarified speech acts as workflow categories, added multimodal scope note, and added minimum memory-governance checklist.
V1.2.7	December 2025	Added Appendix on the linguistics stack (plain language) and added a Word-native "three lanes" visual to reinforce Control/Core/Outside-core.
V1.2.6	December 2025	Added a running vignette example and threaded it through Sections 4–6 to anchor the mapping in a concrete, repeatable scenario.
V1.2.5	December 2025	Smoothed transitions and corrected minor terminology (e.g., "machine learning (ML)"); reduced rhetorical heat; clarified where governance shows up in the implications.
V1.2.4	December 2025	Incorporated reader feedback: improved skimmability and clarified lane attribution; minor tone and terminology normalization.
V1.2.3	December 2025	Plain-spoken sweep: tightened prose for governance audiences while preserving technical accuracy and the three-lane frame.
V1.2.2	December	Added author background caveat as a title footnote to frame the essay as a

	2025	practitioner synthesis rather than a disciplinary contribution.
V1.2.1	December 2025	Added policy-reader on-ramp materials: Reader Guide plus plain-language appendices (technical notes + glossary) to reduce barriers for non-ML audiences.
V1.2.0	December 2025	Added seam failure mode (Control ↔ retrieval conflict) and clarified inference-time boundary-blurring; strengthened governance framing around common ground and auditability.
V1.1.2	December 2025	Structural sharpening for skimmability: clearer section headings, explicit “how to use this” guidance, improved transitions, and table legend/reading instructions.
V1.0.0	December 2025	Initial practitioner draft: mapped the four philosophers to a linguistics stack and then to a three-lane GPT system frame (Control / Core / Outside-core).

Appendix A. Full mapping table

The table below provides the full linguistics-to-GPT mapping referenced in the main text.

Linguistics layer (tags)	CONTROL (message hierarchy)	CORE (transformer)	OUTSIDE-core (runtime + I/O)
Deixis / Indexicals [W,L]	prompt can pin "here/now/this/that" via explicit context ("In this doc…", "In 2025…")	resolves indexicals only via textual cues in-window; brittle when cues are missing	runtime context injection (time, locale, tool outputs) can stabilize reference
Empirical verification / World-tethering [N(LIMIT),W,L,D]	prompts can require sources, quoting, and "don't guess" rules	no inherent truth-check; can generate plausible continuations absent grounding	tools/RAG/browsing (if present) provide external checks; otherwise none
Sociolinguistics / Register [W,L]	system/dev tone, norms; user constraints	learned style/register priors	decoding settings affect variability/verbosity
Conversation / Interaction [W,L]	role tags, chat template, turn packaging	learned dialogue patterns (repair-like)	stop sequences; tool/function protocol; decoding affects responsiveness
Discourse / Text linguistics [L]	delimiting/quoting/scope defines "context"	attention over context window supports coherence (within window)	context mgmt: truncation, memory injection, summarization; KV-cache/runtime constraints
Pragmatics (speaker meaning) [W,L]	intent cues, priority rules, constraints	pragmatics-ish patterns (inconsistent)	policy/safety layers can override stance; tool gating
Speech acts (assert/request/commit/refuse/hedge) [W,L]	explicit act-shaping: "summarize", "argue", "refuse if…", "ask clarifying questions"	learned act-shaped continuations (asserting, hedging, complying) without truth guarantee	policy enforcement/refusals; constrained output channels (JSON/schema), tool calls as "acts"
Semantics (compositional) [W,D]	framing ("summarize" vs "argue") biases read	contextual reps encode meaning-like distinctions; LM head token distribution	constrained decoding (JSON/schema/grammar), logit bias / stop tokens as steering
Syntax (structure) [D]	formatting can make structure easier/harder (lists, code blocks, etc.)	structure-sensitive behavior via attention/position (emergent)	constrained decoding can enforce structure even if core would drift
Lexicon (words/idioms/MWEs)	domain prompts "activate" vocabulary usage	distributed lexical/idiom patterns in weights	vocab is discrete interface the core predicts

[W,L,D]			
Morphology	minimal direct control role	early-layer regularities over subwords	tokenizer ≈ subword segmentation;
[L,D]			detokenizer merges back
Orthography / Graphology	“use bullets/title case” formatting directives	orthographic conventions learned statistically	raw text I/O: punctuation/case/whitespace render
[W,L]			
Phonology	NO MAPPING (plain text)	NO MAPPING (plain text)	NO MAPPING (plain text)
[—]			
Phonetics	NO MAPPING (plain text)	NO MAPPING (plain text)	NO MAPPING (plain text)
[—]			
Historical linguistics / Diachrony	can request “in 17th-century style” (a control prompt)	diachronic facts/patterns may be present but no intrinsic change mechanism	retrieval/tools (if used) can supply dated facts (outside-core freshness)
[W,L]			
Psycholinguistics / Neurolinguistics	NO MAPPING (not a human cognitive/brain arch)	NO MAPPING (not a cognitive/brain substrate)	NO MAPPING
[D,N(LIMIT)]			

Appendix B. Technical notes in plain language

This appendix defines the minimum technical vocabulary needed to use the mapping in the main text. The goal is practical: to clarify what tends to be inside the model versus in surrounding system components.

- **Tokens (what the system actually outputs)**
  A GPT system generates text one small unit at a time. Those units are called tokens. A token can be a whole word, part of a word, or punctuation. The model does not see words directly; it sees token sequences and produces a ranked set of likely next tokens.
- **Tokenization (why words sometimes split oddly)**
  Before text reaches the model, a tokenizer splits it into tokens. This splitting is optimized for efficient computation, not for linguistic elegance. That is why a single word may be broken into pieces. Tokenization sits Outside-core: it is part of the system interface, not the transformer itself.
- **Decoding (how the system chooses among plausible continuations)**
  The model produces many plausible next-token options. Decoding is the procedure that selects which option becomes the output. More conservative decoding reduces variability; more permissive decoding increases variability. Decoding choices can change tone and risk posture without changing the underlying model weights (Outside-core).
- **Context window (why earlier material drops out)**
  A GPT system has a finite amount of prior text it can attend to at once, often called the context window. When a conversation or document exceeds that limit, older material must be truncated or summarized. This is a major source of apparent inconsistency and drift, and it is primarily an Outside-core constraint.
- **Tools, retrieval, and RAG (how systems look things up)**
  Some deployments connect the model to external sources: search, databases, or document retrieval. Retrieval-Augmented Generation (RAG) is a common pattern: the system retrieves passages, inserts them into the context, and then asks the model to answer using those passages. This can improve factual grounding, but retrieval can fail (missing, stale, or irrelevant sources). Retrieval is Outside-core.
- **Policy and safety layers (why refusals and hedging can change)**
  Many systems apply policy checks before or after the model generates text. These checks can block, rewrite, or redirect outputs, and they often shape the system's stance (more cautious, more constrained). For governance readers, the key point is that policy behavior is frequently an Outside-core control surface.
- **Constrained outputs (schemas, formats, and must look like this)**
  Some systems force outputs to follow a format (for example, JSON, a form, or a tool/function signature). This constrains what speech act is being performed: not just 'answer,' but 'answer in a checkable structure.' These constraints typically live Outside-core, though they shape how the Core's token stream is sampled.
- **Seam failures (when Control and retrieval conflict)**
  A recurring failure mode occurs when the system instructs the model to follow strict constraints (Control), but retrieval provides partial or conflicting evidence. The model may smooth the conflict into a fluent answer. Mitigations include: forcing citation-bound answers, requiring clarification when evidence conflicts, and logging retrieved

passages alongside outputs for audit.

Appendix C. Glossary

Term	Plain-language definition
Control	System and prompt packaging: roles, instructions, templates, and constraints that define the task and permitted moves.
Core	The transformer model: produces a distribution over next tokens given the current context.
Outside-core	The surrounding machinery: tokenization, decoding, context management, policy enforcement, and optional tools/retrieval.
Token	A small unit of text used by the model (word piece, whole word, punctuation).
Tokenizer	Software that splits text into tokens before it reaches the model.
Detokenizer	Software that merges tokens back into readable text.
Decoding	The procedure that selects which token is emitted from the model's ranked possibilities.
Context window	The maximum amount of prior text the model can directly attend to at once.
Truncation	Dropping older context when the window is exceeded.
Summarization/memory injection	Condensing older context and reinserting a shorter version to preserve continuity.
RAG / retrieval	Retrieval augmented generation (RAG). Fetching external passages and inserting them into context to improve grounding.
Tool call / function call	A structured request from the system to an external tool (search, database, etc.).
Policy layer	A rule- or model-based mechanism that can block, rewrite, or redirect outputs.
Schema / constrained output	A required structure for the output (e.g., JSON) enforced by the surrounding system.

Appendix D. The linguistics stack used in this essay

This appendix provides plain-language definitions of the linguistics layers referenced in the mapping tables. The intent is practical: to clarify what each layer names, and why it matters when diagnosing behavior in deployed GPT systems.

Scope note: This mapping is written for plain-text deployments. For natively multimodal systems, the phonology/phonetics "no mapping" rows would need to be revisited, and additional layers (prosody, visual grounding) become first-class.

- **Deixis / indexicals:** Words like "this," "that," "here," and "now" that point to context. Example: "this policy" only makes sense if the referenced policy is specified.
- **Discourse:** How multiple sentences and paragraphs hang together over time (topic continuity, references, coherence). Example: pronouns like "it" should keep pointing to the same thing across a section.
- **Pragmatics:** What a sentence is doing in context (implications, priorities, what is taken as given). Example: "If unsure, ask" changes what counts as a good response.
- **Speech acts:** The basic move being made: asserting, requesting, committing, refusing, hedging. Example: "Cite sources" changes an answer from an assertion to an evidence-bound assertion.
- **Semantics:** The content of what is said - roughly, what a sentence claims. Example: "X caused Y" differs from "X correlated with Y."
- **Syntax:** Sentence structure - how words are arranged to form a well-formed statement. Example: list structure can make constraints easier to follow.
- **Lexicon:** Word choice, idioms, and domain vocabulary. Example: "common ground" vs "shared assumptions" can signal different audiences.
- **Morphology:** Word parts (prefixes/suffixes) and how words are built. In GPT systems this loosely relates to subword pieces produced by tokenization.
- **Orthography / graphology:** Written form: punctuation, casing, headings, spacing, and formatting. Example: bulleting and headings can change readability and compliance.
- **Phonology / phonetics:** Sound-based layers of language. For plain-text GPT systems, these have no direct mapping, except indirectly through spelling patterns in training data.

In this essay's mapping, the higher layers (discourse, pragmatics, and speech acts) are most useful for governance because they describe what an output does in context, not just what it says. The three-lane view then locates where to intervene: Control shapes permitted moves, the Core generates candidate text, and Outside-core provides formatting, verification, and enforcement.

Appendix E: Further Reading

This essay is a practitioner mapping, not a literature review. The sources below are offered as anchors for readers who want to verify the background claims or go deeper.

1. GPT / LLM system mechanics (Core + Outside-core)

    - Transformer foundations (the Core model family)
      Vaswani, A., Shazeer, N., Parmar, N., et al. (2017). Attention Is All You Need.
      Why it matters: establishes the transformer architecture that underlies modern LLMs.
    - Subword tokenization (why "morphology" only weakly maps to tokens)
      Sennrich, R., Haddow, B., & Birch, A. (2016). Neural Machine Translation of Rare Words with Subword Units.
      Why it matters: introduces widely used subword approaches (e.g., BPE) that shape how text is split and represented.
    - Decoding choices (why style, verbosity, and "feel" can change without changing the model)
      Holtzman, A., Buys, J., Du, L., Forbes, M., & Choi, Y. (2019). The Curious Case of Neural Text Degeneration.
      Why it matters: shows how sampling/decoding strategies materially affect generated text.
    - Retrieval-Augmented Generation (world-tethering via external sources)
      Lewis, P., Perez, E., Piktus, A., et al. (2020). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks.
      Why it matters: explains how retrieval + generation can improve factuality/provenance compared to generation alone.
    - Plain-language grounding for non-specialists (tokens, language modeling, context limits)
      Jurafsky, D., & Martin, J. H. Speech and Language Processing (draft / online).
      Why it matters: a clear, widely used reference for the NLP basics that sit behind "next-token prediction," context windows, and core terminology.
    - Inference-time optimization (optional, if "KV-cache" or serving costs come up)
      Raschka, S. (practitioner explainer). Understanding and Coding the KV Cache in LLMs from Scratch.
      Why it matters: explains a common inference-time optimization that affects latency/memory, even though it doesn't change "what the model is."
    - Modern LLM architecture variation
      Raschka, S. LLM Architecture Gallery.
      Why it matters: provides a current practitioner-oriented overview showing that contemporary LLMs vary substantially in attention, routing, normalization, and architectural details; useful context for why "Core" is a governance abstraction rather than a simple architectural claim.

2. Philosophical anchors (interpretive constraints)

    - Wittgenstein (meaning in use; language-games)
      Wittgenstein, L. Philosophical Investigations. (esp. remarks commonly cited around "meaning as use")
      Why it matters: frames meaning as grounded in use and practice, not as a hidden mental object.
    - Lewis (convention; coordination; common knowledge; scorekeeping)
      Lewis, D. Convention: A Philosophical Study.
      Lewis, D. "Scorekeeping in a Language Game."
      Why it matters: provides tools for thinking about shared expectations, common ground, and conversational bookkeeping across turns.
    - Dennett (the intentional stance as a pragmatic interpretive tool)
      Dennett, D. The Intentional Stance.
      Why it matters: explains why "as if" agency talk can be predictive—and why it becomes a trap when treated as literal ontology.
    - Nagel (the boundary on subjective experience claims)
      Nagel, T. (1974). "What Is It Like to Be a Bat?"
      Why it matters: clarifies why fluent first-person style does not warrant claims about inner experience.

3. Quick orientation references

    - Stanford Encyclopedia of Philosophy (SEP) entries
      Entries on Wittgenstein, Lewis, Dennett/intentional stance, and related topics.
      Why it matters: concise, authoritative summaries for readers who want a reliable overview before going to primary texts.