# CONTEXT ENGINEERING: AN INTELLECTUAL GENEALOGY OF THE VOID MANAGEMENT THESIS

## 1. APEX: THE GENEALOGICAL CLAIM

The discipline of Context Engineering, crystallizing in the mid-2020s as the successor to the ad-hoc practice of prompt engineering, represents a profound convergence of intellectual traditions spanning over two millennia. While contemporary discourse frames it as a technical optimization of Large Language Model (LLM) inference—the "delicate art and science of filling the context window," as defined by Andrej Karpathy 1—a rigorous genealogical analysis reveals it to be the re-emergence of fundamental questions regarding situated cognition, semiotic density, and systems theory. The central claim of this report is that **Context Engineering is the convergence of three distinct lineages: the Situated Cognition tradition (Suchman, Hutchins), the Thick Description tradition (Geertz, Bateson), and the pragmatic engineering of the Prompt (Karpathy, Lütke).**

This convergence culminates in the **Void Management Thesis**, a design synthesis that treats the context window not merely as a passive receptacle for text, but as a bounded, material resource—a "jar" that organizes the wilderness of information. The thesis asserts that intelligence is not an inherent property of the model's weights but an emergent property of the interaction between the model and its engineered context. The limitations of the medium—the finiteness of the window, the cost of the token, the decay of attention—are the very constraints that enable meaning to emerge.

This document traces the intellectual lineage of this thesis, moving from the present-day synthesis back through the convergence of the late 20th century, the origins in mid-century cybernetics, and finally to the prehistoric philosophical roots in Aristotelian ethics and phenomenology. It serves as a self-contained intellectual history for the context engineer, grounding technical practice in theoretical depth.

## 2. LEVEL 1: SYNTHESIS (Void Management as Culmination)

The Void Management Thesis is the contemporary design synthesis of fifty years of context research. It posits that the primary task of the AI engineer is no longer the training of the model (the creation of static weights) but the curation of the dynamic void—the active, inference-time environment in which intelligence is situated.

### The Materiality of Context

The foundational axiom of Void Management is that **context is material**. In the computational realm of the LLM, context has weight, cost, decay rate, and rigid engineering constraints. It is not an abstract "meaning" but a physical state of the system, measured in tokens and processed in GPU memory.

#### The Economics of the Token

As Andrej Karpathy articulated in 2025, the context window functions as the Random Access Memory (RAM) of the cognitive operating system, while the LLM serves as the Central Processing Unit (CPU).1 In this architecture, the token is the fundamental economic unit. Every token introduced into the "void" (the context window) incurs a cost in compute cycles (attention is quadratic in many architectures) and, crucially, in "attention budget."

Modern research indicates that as context grows, models tend to allocate attention to immediate information at the expense of parametric knowledge or complex reasoning.1 This phenomenon, often termed "context distraction" or "lost in the middle," underscores the materiality of the void. It is a finite resource where information competes for survival. The "pressure" on the context window is a quantifiable metric of the cognitive load placed on the system.

The code snippet referencing L0: Void Management and self.pressure = 0.0 2 provides a technical reification of this theoretical stance. Here, "pressure" is likely a calculated value derived from the ratio of used tokens to the total window size, or perhaps a measure of the semantic divergence within the context. "Void Management," in this technical implementation, is the algorithmic regulation of this pressure—ensuring that the density of information remains within the "habitable zone" for the model's reasoning capabilities.

#### The Mining Metaphor: Structural Integrity

The term "Void Management" is borrowed from the mining industry, where it refers to the management of underground spaces (stopes) created by the extraction of ore.3 In mining, a void is a structural risk; if not managed (e.g., by backfilling), it can lead to "stope failure" or collapse.

The metaphor is strikingly precise for Context Engineering. The "void" of the context window is created by the "extraction" of the user's intent. If this void is not carefully managed—if it is filled with "garbage" or left structurally unsound—the result is "cognitive collapse," manifested as hallucination. The "geotechnical assessment" required in mining 4 maps to the "eval pipelines" of context engineering.5 Just as a mine manager must assess the stability of the rock face, the context engineer must assess the semantic stability of the prompts and retrieved documents. A "failed stope" in a mine exposes personnel to risk; a "failed context" in an AI agent exposes the user to erroneous actions or security vulnerabilities (context poisoning).6

#### The Housing Metaphor: Process and Turnover

A second, equally relevant source for the "Void Management" term comes from social housing administration.7 In this domain, void management is the process of managing a property between tenancies—from the termination of one tenant to the re-letting to another. It involves cleaning, repairing, and ensuring the space is "habitable."

In the lifecycle of an AI agent, particularly a long-running one, "void management" refers to the "context clearing" or "state reset" that must occur between tasks. When an agent finishes a complex coding task (Tenant A) and moves to a creative writing task (Tenant B), the context window must be "cleaned." Residual tokens from the previous task (the "trash" left by the previous tenant) can contaminate the new inference, leading to "mode confusion" or privacy leaks. The "Void Management Policy" of a housing authority 9 finds its parallel in the "Context Retention Policy" of an AI system—defining what is kept (long-term memory), what is archived (to disk/vector DB), and what is deleted.

### Design Principle: The Jar and the Wilderness

The central design principle of the Void Management Thesis is the distinction between the **Wilderness** and the **Jar**.

* **The Wilderness:** This is the vast, unorganized ocean of information available to the system—the "Knowledge Context".1 It includes the entirety of the internet, the user's complete file history, and the millions of vectors in a database. It is "thick" in the sense of being voluminous, but "thin" in the sense of being unorganized. It is the "background" (Winograd) or the "field" (Garfinkel).
* **The Jar:** This is the context window itself. It is a bounded, rigid container. It has a hard limit (e.g., 128k tokens).

The act of engineering is the act of filling the jar from the wilderness. The "Void Management" thesis asserts that **unbounded context is unmanageable context**.10 If one attempts to pour the entire wilderness into the jar, the system overflows (out of memory) or becomes diluted (attention dispersion).

The "Jar" organizes the wilderness by imposing structure. This structure is often a "grid"—a schema, a template, or a set of few-shot examples.10 The "grid" allows for **fixed structure with variable instantiation**. The prompts (the structure of the jar) remain constant; the retrieved data (the contents) flow through it. This aligns with the "Offloading" strategy described by the Manus team 1, where the file system acts as unlimited external memory (the wilderness), achieving 100:1 compression ratios. Only the "summary" or "metadata" enters the jar. The key insight is **reversibility**: one can always retrieve the full content from the wilderness using the reference in the jar, but one cannot fit the wilderness in the jar.

### Historical Position

The Void Management Thesis sits at the apex of the pyramid because it synthesizes the insights of the past into a pragmatic discipline for the future. It is where **Situated Cognition** (the understanding that intelligence is local) meets **Prompt Engineering** (the technical capability to manipulate the locus of intelligence). It validates **Thick Description** by proving that "meaning" requires a dense, curated context, and it operationalizes **Systems Thinking** by treating the context window as a feedback-controlled system.

## 3. LEVEL 2: CONVERGENCE (1990s-2020s)

The transition from the 20th to the 21st century marked a convergence of three distinct streams of thought: the computational pragmatism of modern AI, the interaction design philosophy of the 1980s, and the critique of rationalism in computer science. This convergence laid the immediate groundwork for Context Engineering.

### Karpathy’s Prompt Engineering (2023-2025)

The most visible vector of this convergence is the rapid evolution of "prompt engineering" into "context engineering." In the early 2020s, with the advent of GPT-3, the focus was on "prompting"—the art of asking the right question.1 By 2025, industry leaders like Andrej Karpathy and Tobi Lütke crystallized the shift.

The Shift: "Context engineering is the new prompt engineering".1

The Mechanism: Karpathy’s formulation of the LLM as an Operating System provided the necessary mechanistic metaphor.1

* **CPU:** The LLM (weights + inference engine).
* **RAM:** The Context Window (working memory).
* **Disk:** Vector Database / File System (long-term storage).

This metaphor legitimized the "void" as a programmable space. It moved the discipline from "whispering" to the model (a mystical art) to "memory management" (an engineering science). The "delicate art and science of filling the context window" 1 is fundamentally an optimization problem: maximizing the *signal-to-noise ratio* of the tokens in RAM.

**Evidence of Convergence:**

* **Tobi Lütke (June 19, 2025):** "Context engineering is the art of providing all the context for the task to be plausibly solvable by the LLM".13
* **Andrej Karpathy (June 25, 2025):** "Context engineering is the delicate art and science of filling the context window with just the right information for the next step".13
* **Harrison Chase (June 23, 2025):** "Context engineering is building dynamic systems to provide the right info".13

This synchronization of terminology among industry leaders indicates a "paradigm shift" (Kuhn) where the community collectively recognized that the "Prompt" was too narrow a unit of analysis. The "Context"—the entire system state—became the new atomic unit.

### Winograd & Flores: The Breakdown of Rationalism (1986)

Decades prior to Karpathy, Terry Winograd and Fernando Flores published *Understanding Computers and Cognition*.15 Their work, deeply influenced by Heideggerian phenomenology, launched a critique of the "rationalistic tradition" in AI—the idea that intelligence consists of manipulating symbolic representations of the world.

Computers as Tools for Conversation:

They argued that computers are fundamentally tools for conversation and action, situated within a social context.15 This prefigures the "Chat" interface of modern LLMs. The "conversation" is not just text exchange; it is the generation of commitments and actions.

Breakdown:

Winograd and Flores introduced the concept of "breakdown." Tools are "ready-to-hand" (invisible) when functioning smoothly, but become "present-at-hand" (visible objects of study) when they fail or "break down".17

* **Ready-to-hand:** When an LLM agent works perfectly, the user ignores the prompt and focuses on the result. The context is transparent.
* **Breakdown:** A "hallucination" is a breakdown. It is a moment where the seamless flow of context is ruptured, revealing the artificiality of the system.

Design Implications:

Their assertion that "design is ontological"—that designing tools designs new ways of being—prefigures the Void Management view that engineering context is engineering the being of the AI agent. The context engineer does not just supply data; they structure the "blindness" and "sight" of the model, determining what is obvious and what is invisible.17 By filtering the wilderness, the engineer decides what constitutes the "world" for the agent.

### Norman’s Affordances and Constraints (1988)

Donald Norman’s *The Psychology of Everyday Things* (later *The Design of Everyday Things*) provided the third pillar of this convergence.18

Affordances:

Norman popularized the concept of "affordances"—the action possibilities perceived by an actor in an environment. In the physical world, a door plate affords pushing. In the digital void of an LLM, a structured JSON schema affords precise data extraction. A conversational, open-ended prompt affords creative rambling.

The context engineer designs the affordances of the window. If the engineer wants the model to use a tool, they must provide a tool definition (affordance) that is "visible" and "perceptible" to the model.20

Constraints:

Norman emphasized that constraints are as important as affordances. Constraints limit behavior to prevent error.

* **Physical Constraints:** The size of the context window (e.g., 128k tokens).
* **Semantic Constraints:** Instructions like "Answer only in JSON."
* **Cultural Constraints:** The "persona" or tone instructions.

The "Void Management" thesis adopts Norman’s view: the wilderness is full of false affordances (irrelevant data); the engineered context must *constrain* the model to the path of success. Meaning emerges from these constraints.

### Convergence Point

**All three traditions recognize that meaning emerges from constraints, not from unbounded possibility.**

* **Karpathy:** Bounded memory (RAM) forces prioritization of tokens.1
* **Winograd:** Bounded situation ("thrownness") forces localized action.15
* **Norman:** Bounded interface forces correct usage.18

The Void Management thesis is the practical application of this tripartite realization.

## 4. LEVEL 3: TRADITIONS (Three Lineages)

Beneath the immediate convergence of the modern era lie three deep intellectual traditions that provide the theoretical substance for Context Engineering: Situated Cognition, Thick Description, and Systems Thinking.

### Tradition 1: Situated Cognition

The Situated Cognition tradition, emerging in the 1980s and 1990s, fundamentally challenged the computational theory of mind. It argued that thinking is not a process that happens "inside the head" (or the CPU) but is inextricably linked to the physical and social context.

#### Suchman: Plans and Situated Actions (1987)

Lucy Suchman’s seminal work 21 dismantled the idea that human action is driven by abstract "plans" executed by a cognitive processor. Through her ethnographic study of Xerox photocopier usage, she demonstrated that plans are not controlling programs but distinct **resources** used by actors to account for their actions *after* or *during* the fact. Action is "situated"—it is an improvisation based on the immediate material circumstances.

Implication for Context Engineering:

This is a radical insight for AI. It implies that the "System Prompt" (the plan) is not a rigid controller of the LLM. Instead, the "situated action" of the model depends on the immediate tokens present in the window (the situation). The model does not "follow" the plan in a deterministic sense; it uses the plan as a resource to navigate the current context.

* **Context Drift:** This explains why context drift occurs. As the "situation" (the conversation history) grows and changes, it exerts more gravitational pull on the model's actions than the abstract plan (system prompt).1 The immediate situation *is* the controller.
* **Link to Void:** The bounded space of the window *is* the plan. The engineer must ensure that the "situation" (the tokens in the window) always aligns with the desired "plan."

#### Lave: Cognition in Practice (1988)

Jean Lave’s research on arithmetic in everyday life (e.g., grocery shopping) showed that people do not use abstract school-taught math algorithms in the wild.24 Instead, they use the environment (the context) to perform the calculation—for example, physically comparing package sizes rather than calculating price-per-ounce mentally.

Implication for Context Engineering:

This validates the "Offloading" strategy.1 Just as Lave’s shoppers offloaded cognitive effort into the physical arrangement of groceries, context engineers offload token-heavy tasks to external tools (calculators, code interpreters), bringing only the result into the cognitive focus of the model.

* **Cognition is Distributed:** Thinking happens between the weights and the window. The "intelligence" of the system is not just in the model; it is in the *interaction* between the model and the engineered context.

#### Hutchins: Cognition in the Wild (1995)

Edwin Hutchins extended this to "distributed cognition".26 His study of ship navigation teams revealed that the "intelligence" of the ship was not in the captain’s head but distributed across charts, alidades, and the communication protocols of the crew.

Implication for Context Engineering:

The "agent" is a distributed cognitive system.

* **Cognitive Artifacts:** The charts and rulers of the ship map to the **"few-shot examples"** and **"structured templates"** in the context window.10 These are artifacts that guide reasoning.
* **Propagation of State:** Hutchins described how a navigational state propagates through different representations (from landmark to bearing to line on a chart). Similarly, a context engineer designs the propagation of state from "User Query" to "Search Term" to "Retrieved Document" to "Final Answer".28

### Tradition 2: Thick Description

If Situated Cognition tells us *where* thinking happens, Thick Description tells us *what* makes information meaningful.

#### Geertz: The Interpretation of Cultures (1973)

Clifford Geertz coined the term "thick description" to distinguish between the raw physical observation of an act (a "twitch") and its cultural meaning (a "wink").29 A thin description records the contraction of the eyelid; a thick description records the conspiracy, the satire, or the flirtation.

Implication for Context Engineering:

In the era of Prompt Engineering, "thin description" was the norm: short, direct instructions ("Write a poem"). Context Engineering demands "thick description."

* **The Grid Preserves Thickness:** To get a high-quality output from an LLM, the engineer must provide the "cultural" context: the persona, the tone, the constraints, the history, and the intent.5 The "grid" of the context window must be populated with enough "thickness" (examples, behavioral guidelines, nuances) to allow the model to distinguish a twitch from a wink.
* **Addressability:** The challenge is to make this thickness *addressable* by the model. Unstructured thickness is noise. Structured thickness (Thick Description via JSON/XML) is signal.

#### Bateson: Steps to an Ecology of Mind (1972)

Gregory Bateson defined information as "**a difference that makes a difference**".32 In a system of infinite data (the wilderness), most differences make no difference—they are noise.

Implication for Context Engineering:

This is the fundamental heuristic for "pruning" the context.34 When filling the "jar," every piece of information must be scrutinized: does this token constitute a "difference that makes a difference" to the model's output? If not, it is noise that degrades the system's performance (increasing pressure without increasing signal).

* **Schismogenesis:** Bateson’s concept of "schismogenesis" (feedback loops that lead to breakdown) warns of the dangers of self-reinforcing errors in LLM dialogue loops. A small hallucination, if fed back into the context, can amplify into a complete divergence from reality.33

### Tradition 3: Systems Thinking

The third lineage provides the mechanisms for control and stability.

#### Meadows: Thinking in Systems (2008)

Donella Meadows taught that systems are defined by their **boundaries** and their **feedback loops**.35 A system without feedback cannot self-correct.

**Implication for Context Engineering:**

* **Leverage Points:** Meadows’ concept of "leverage points"—places in a system where a small change produces a large effect—is the holy grail of context optimization. A single instruction in the System Prompt (a high-leverage point) can alter the behavior of the entire agent more effectively than thousands of tokens of few-shot examples (a lower leverage point).36
* **The Bound is the System:** "81 cells is the bounded system" refers to the concept that a system is defined by what is *inside* the boundary. In Void Management, the 128k token window (or similar limit) defines the universe of the agent. What is outside does not exist for the agent.

#### Ashby: Introduction to Cybernetics (1956)

Ross Ashby’s **Law of Requisite Variety** states that "only variety can destroy variety".37 To control a system, the controller must have as many states as the system being controlled.

Implication for Context Engineering:

This explains the failure of simple prompts for complex tasks. A complex real-world task (high variety) cannot be controlled by a simple, low-variety prompt. The context window must contain "requisite variety"—enough tools, knowledge, and contingency plans—to match the complexity of the user's request.

* **Control through Constraint:** The Void Management thesis is essentially the management of Requisite Variety within the constraints of the context window.

## 5. LEVEL 4: ORIGINS (1970s-1980s)

The traditions of the 1990s did not emerge in a vacuum; they were built upon the foundational work of the 1970s and 80s in cybernetics, ethnomethodology, and activity theory.

### Cybernetics (Wiener, Ashby)

Cybernetics, the study of control and communication in the animal and the machine, is the grandfather of Context Engineering. Norbert Wiener and Ross Ashby established the principles of feedback and control that underpin modern "agentic" AI.37

Control through Constraint:

The central cybernetic insight is that control is achieved through constraint. A thermostat controls temperature not by "understanding" heat, but by constraining the system within boundaries. Similarly, context engineering does not teach the LLM to "understand" in a human sense; it controls the model's probabilistic generation by imposing constraints (context) that narrow the search space of the next token. The "Void" is the state space; the context engineer erects the cybernetic boundaries that keep the trajectory of the system within the desired region (the "attractor").

### Ethnomethodology (Garfinkel)

Harold Garfinkel’s *Studies in Ethnomethodology* (1967) introduced the concept of **"indexicality"**—the idea that the meaning of any word or action is indexed to its specific, local context.39 There is no context-free meaning.

Meaning is Locally Produced:

For LLMs, this is a literal truth. The meaning of a token is mathematically derived from its attention to every other token in the sequence.

* **Accountability:** Garfinkel’s work on "accountability"—how members of a society make their actions visibly rational and reportable—is crucial for AI transparency. A well-engineered context forces the agent to produce a "Chain of Thought" (an account) that makes its reasoning visible and indexical. The "grid" of the Void Management system preserves the indexical links between the user's intent and the model's action.

### Activity Theory (Vygotsky, Leontiev)

Activity Theory, rooted in Soviet psychology, offers the most robust model for tool use. Lev Vygotsky 41 and A.N. Leontiev 42 argued that human cognition is mediated by tools (both physical tools and psychological tools like language).

Zone of Proximal Development (ZPD):

Vygotsky proposed the ZPD as the distance between what a learner can do alone and what they can do with help.43

* **Context as Scaffolding:** Context Engineering effectively places the LLM in a ZPD. The "System Prompt" serves as the "more knowledgeable other," scaffolding the model's performance. The model, which might fail at a complex task in isolation, succeeds when supported by the scaffolding of a rich context.

Hierarchical Structure of Activity:

Leontiev distinguished between "activity" (motivated by a need), "action" (directed at a goal), and "operation" (automatic, unconscious execution).42

* **Mapping to Agents:** Context engineering organizes the LLM's work into these levels.
  + **Activity:** The user's high-level intent (e.g., "Write a novel").
  + **Action:** The specific tool calls (e.g., "Generate outline," "Research setting").
  + **Operation:** The token generations (the automatic output).
  + **Failure Mode:** The failure of "prompt engineering" was often a failure to distinguish between these levels—giving an operation-level instruction for an activity-level problem.

## 6. LEVEL 5: PREHISTORY (Before Computers)

The deepest roots of Context Engineering tap into the bedrock of Western philosophy. The problems of context, practical wisdom, and the relationship between the general rule and the specific case were articulated long before the invention of the microchip.

### Aristotle — Techne vs Phronesis

In the *Nicomachean Ethics*, Aristotle distinguishes between different intellectual virtues.45

* **Episteme:** Scientific knowledge, universal and invariable. (The "weights" of the model—parametric knowledge).
* **Techne:** Craft or art, the skill of making things, governed by rules. (The code and infrastructure of the AI).
* **Phronesis:** Practical wisdom. The ability to discern the right course of action in a *specific, changing situation* where the rules do not perfectly apply.

Context Engineering as Computational Phronesis:

Context Engineering is the attempt to imbue a machine of Episteme and Techne with a simulation of Phronesis. A "prompt" is often a request for a general rule (Episteme). But a complex agentic task requires Phronesis—navigating the "particulars" of the user's messy, contradictory context. The "Void Management" thesis is a design for Phronesis: it creates a space where the particulars (the contents of the jar) can weigh against the universals (the model's training).

### Rhetoric (Aristotle, Cicero)

The classical tradition of Rhetoric (Aristotle, Cicero) is the study of context-dependent communication.47

* **Kairos:** The opportune moment, the "right time" to speak. In Context Engineering, this is the "dynamic" injection of information—ensuring the model has the information *exactly when it is relevant*, not before (distraction) and not after (irrelevance).
* **Decorum:** The fitness of the speech to the occasion. A "System Prompt" that defines a persona is essentially an instruction in *Decorum*—telling the model what tone and style befit the current "occasion" (context).

### Phenomenology (Husserl, Heidegger)

Martin Heidegger’s *Being and Time* 49 provides the ultimate ontological grounding.

* **Thrownness (Geworfenheit):** Heidegger argued that *Dasein* ("Being-there") is always already "thrown" into a world. We are never outside of context.
* **The LLM as Cartesian Subject:** The LLM in its base state is a Cartesian subject—a brain in a vat, disconnected from the world.
* **Context Engineering as "Throwing":** Context Engineering is the process of "throwing" the LLM into a world. By filling the context window with a specific history, a specific user profile, and specific tools, the engineer gives the model a *Dasein*. The model becomes "Situated" (Suchman) because it is "Thrown" (Heidegger) into a "World" (Context). The "Void" is the space where the World is constructed.

### James C. Scott: Seeing Like a State

Finally, the work of James C. Scott in *Seeing Like a State* 51 provides a political-philosophical lens.

* **Techne vs. Metis:** Scott contrasts *Techne* (state-imposed, simplified, standardized knowledge) with *Metis* (local, practical, cunning intelligence). High Modernist states try to make the world legible by imposing *Techne* and erasing *Metis*—often leading to disaster (e.g., scientific forestry failing because it ignored the complex ecology).
* **LLMs as High Modernism:** LLMs are High Modernist engines of *Techne*. They are trained on the "average" of human knowledge, flattening local nuance.
* **Context Engineering as Metis:** Context Engineering is the re-injection of *Metis*. By filling the context with the specific, messy details of the user's local reality (the "thick description"), the engineer protects the user from the "blindness" of the state-like model. The Void Management thesis is a strategy for preserving *Metis* inside the machine of *Techne*.

## 7. ARGUMENT EXTRACTION

### For Intellectual History:

"Context engineering is the convergence of situated cognition, thick description, and prompt engineering. The void management thesis is where Suchman meets Karpathy."

### For Theoretical Claims:

"All three traditions recognize that meaning emerges from constraints. Plans, affordances, and prompts all work by bounding possibility, not expanding it."

### For Design Grounding:

"From cybernetics: control through constraint. From ethnomethodology: meaning locally produced. From activity theory: tools mediate cognition. The grid synthesizes all three."

## 8. INTAKE GRID FOR GENEALOGICAL SOURCES

The following grid organizes the primary sources referenced in this genealogy, mapping them to the levels of the Argument Pyramid.

| **Era** | **Look For** | **Maps To** | **Key Figures** | **Key Concepts** |
| --- | --- | --- | --- | --- |
| **2020s** | Prompt/Context Engineering | **CONVERGENCE** | Karpathy, Lütke, Chase | Token Economics, CPU/RAM Metaphor, Void Management |
| **1990s** | Situated Cognition, Distributed Cognition | **TRADITIONS** | Suchman, Hutchins, Lave | Plans as Resources, Distributed Cognition, Cognitive Artifacts |
| **1980s** | HCI, Design Theory | **CONVERGENCE** | Winograd, Flores, Norman | Breakdown, Ready-to-hand, Affordances, Constraints |
| **1970s** | Thick Description, Cybernetics, Systems | **TRADITIONS / ORIGINS** | Geertz, Bateson, Meadows | Thick Description, Ecology of Mind, Leverage Points |
| **1960s** | Ethnomethodology, Activity Theory | **ORIGINS** | Garfinkel, Vygotsky, Leontiev | Indexicality, Accountability, ZPD, Activity/Action/Operation |
| **Pre-Comp** | Philosophy, Rhetoric | **PREHISTORY** | Aristotle, Heidegger, Scott | Phronesis, Techne, Metis, Thrownness, Kairos |

### Table of Comparisons: Thin vs. Thick Context

| **Concept** | **Thin Description (Prompt Engineering)** | **Thick Description (Context Engineering)** | **Theoretical Basis** |
| --- | --- | --- | --- |
| **Goal** | Elicit a specific response. | Maintain a consistent world/state. | Winograd (Ontology) |
| **Input** | A single instruction ("Write a poem"). | A bounded system (Persona + History + Tools). | Ashby (Requisite Variety) |
| **Meaning** | Dictionary definition (Semantic). | Locally produced (Indexical). | Garfinkel (Indexicality) |
| **Error** | "Wrong answer." | "Hallucination" / "Breakdown." | Heidegger (Breakdown) |
| **Control** | Tweaking words ("Whispering"). | Managing constraints ("bounding the void"). | Norman (Constraints) |
| **Metaphor** | Asking a question. | Managing a mine / Managing a tenancy. | Void Management (Mining/Housing) |

*This genealogy is self-contained. The lineage traces a clear path from the ancient question of "how to act wisely in a specific situation" to the modern engineering challenge of "how to fill the context window for the next token."*

#### Works cited

1. Deep Dive into Context Engineering for Agents - Galileo AI, accessed December 10, 2025, <https://galileo.ai/blog/context-engineering-for-agents>
2. 4qx-holarchy/can-you-analyse-and-report-on-1739483038575, accessed December 10, 2025, <https://code.organicdesign.nz/organicdesign/4qx-holarchy/src/commit/ad69f4ebe0973bd2e824901ee75de08898875926/context/oracle-conversations/can-you-analyse-and-report-on-1739483038575-export.txt>
3. Underground Operators Conference 2025 Proceedings - AusIMM, accessed December 10, 2025, <https://www.ausimm.com/globalassets/conferences-and-events/underground-operators-2027/underground-operators-2025/ugops-2025-proceedings-ebook-2.pdf>
4. WILPINJONG COAL PROJECT OPEN CUT OPERATIONS MINING OPERATIONS PLAN 2019 – 2020 - Peabody Energy, accessed December 10, 2025, <https://www.peabodyenergy.com/Peabody/media/MediaLibrary/Operations/Australia%20Mining/New%20South%20Wales%20Mining/Wilpinjong%20Mine/20180117_MOP_Open-Cut_2019-2020_Amendment-A_FINAL.pdf>
5. Context Engineering Guide, accessed December 10, 2025, <https://www.promptingguide.ai/guides/context-engineering-guide>
6. A Gentle Introduction to Context Engineering in LLMs - KDnuggets, accessed December 10, 2025, <https://www.kdnuggets.com/a-gentle-introduction-to-context-engineering-in-llms>
7. (Public Pack)Agenda Document for Cabinet, 24/07/2024 14:00 - Meetings, agendas, and minutes - Stevenage Borough Council, accessed December 10, 2025, <https://democracy.stevenage.gov.uk/documents/g5703/Public+reports+pack+Wednesday+24-Jul-2024+14.00+Cabinet.pdf?T=10>
8. 150 Best learning officer Jobs in , London (November 2025) | JOB, accessed December 10, 2025, <https://jobtoday.com/gb/jobs-learning-officer/london_near_>
9. (Public Pack)Agenda Document for Council, 23/04/2025 18:00, accessed December 10, 2025, <https://democracy.middevon.gov.uk/documents/g1972/Public%20reports%20pack%2023rd-Apr-2025%2018.00%20Council.pdf?T=10>
10. davidkimai/Context-Engineering: "Context engineering is the delicate art and science of filling the context window with just the right information for the next step." — Andrej Karpathy. A frontier, first-principles handbook inspired by Karpathy and 3Blue1Brown for moving beyond prompt engineering to the wider discipline of context design, orchestration - GitHub, accessed December 10, 2025, <https://github.com/davidkimai/Context-Engineering>
11. What Is Context Engineering? A Guide for AI & LLMs | IntuitionLabs, accessed December 10, 2025, <https://intuitionlabs.ai/articles/what-is-context-engineering>
12. Beyond prompt engineering: the shift to context engineering | Nearform, accessed December 10, 2025, <https://nearform.com/digital-community/beyond-prompt-engineering-the-shift-to-context-engineering/>
13. What is Context Engineering, Anyway? - Zep, accessed December 10, 2025, <https://blog.getzep.com/what-is-context-engineering/>
14. Context Engineering: The Future of AI Systems | by Meghana Harishankara | Medium, accessed December 10, 2025, <https://medium.com/@meghanaharishankara/context-engineering-the-future-of-ai-systems-a52062c727f0>
15. Understanding computers and cognition | Semantic Scholar, accessed December 10, 2025, <https://www.semanticscholar.org/paper/Understanding-computers-and-cognition-Winograd-Flores/0ee153002e00e9a52eadac3ab0c93a5bd0582a8f>
16. Terry Winograd home page - Publications - Google Sites, accessed December 10, 2025, <https://sites.google.com/view/terrywinogradhomepage/home/publications>
17. T. Winograd and F. Flores, Understanding Computers and Cognition: A New Foundation for Design (Ablex, Norwood, NJ, 1986), accessed December 10, 2025, <https://billclancey.name/Review-Winograd-Flores-Clancey-AIJ1987.pdf>
18. Norman, D. (1988). The Psychology of Everyday Things. New York Basic Books, 140. - References - Scientific Research Publishing, accessed December 10, 2025, <https://www.scirp.org/reference/referencespapers?referenceid=1487772>
19. The Design of Everyday Things - ICDST E-print archive of engineering and scientific PDF documents, accessed December 10, 2025, <https://dl.icdst.org/pdfs/files4/4bb8d08a9b309df7d86e62ec4056ceef.pdf>
20. The Psychology of Everyday Things - Donald A. Norman - Google Books, accessed December 10, 2025, <https://books.google.com/books/about/The_Psychology_of_Everyday_Things.html?id=OlNSRAAACAAJ>
21. Human-Machine Reconfigurations: Plans and Situated Actions: 2nd Edition - ResearchGate, accessed December 10, 2025, <https://www.researchgate.net/publication/265092509_Human-Machine_Reconfigurations_Plans_and_Situated_Actions_2nd_Edition>
22. [PDF] Plans and Situated Actions: The Problem of Human-Machine Communication (Learning in Doing: Social, | Semantic Scholar, accessed December 10, 2025, <https://www.semanticscholar.org/paper/Plans-and-Situated-Actions%3A-The-Problem-of-in-Suchman/5416463537f8c6be1199951b4fd6f8d5dae14920>
23. Plans and situated actions : the problem of human-machine communication : Suchman, Lucille Alice : Free Download, Borrow, and Streaming - Internet Archive, accessed December 10, 2025, <https://archive.org/details/planssituatedact0000such>
24. Cognition in Practice: Mind, Mathematics and Culture in Everyday Life - About Google Books, accessed December 10, 2025, <https://books.google.kg/books?id=-JD6ngEACAAJ&source=gbs_book_other_versions_r&cad=3>
25. Cognition in Practice: Mind, Mathematics and Culture in Everyday Life - Google Books, accessed December 10, 2025, <https://books.google.com/books/about/Cognition_in_Practice.html?id=n6eiH3iPVKYC>
26. Cognition in the wild / Edwin Hutchins | Catalogue | National Library of Australia, accessed December 10, 2025, <https://catalogue.nla.gov.au/catalog/580835>
27. Cognition in the Wild, Edwin Hutchins - EUSSET Digital Library, accessed December 10, 2025, <https://dl.eusset.eu/items/b0028a89-c1c2-430c-96c5-7066fa157e66>
28. ‪Edwin Hutchins‬ - ‪Google Scholar‬, accessed December 10, 2025, <https://scholar.google.com/citations?user=T5WQ2EcAAAAJ&hl=en>
29. Geertz, C. (1973). Thick Description Toward an Interpretive Theory of Culture. Basic Books. - References - Scientific Research Publishing, accessed December 10, 2025, <https://www.scirp.org/reference/referencespapers?referenceid=3712718>
30. What is Thick Description in Qualitative Research? - QDAcity, accessed December 10, 2025, <https://qdacity.com/thick-description/>
31. Context Engineering: Techniques, Tools, and Implementation - iKala, accessed December 10, 2025, <https://ikala.ai/blog/ai-trends/context-engineering-techniques-tools-and-implementation/>
32. Bateson, G. (1972). Steps to an Ecology of Mind Collected Essays in Anthropology, Psychiatry, Evolution, and Epistemology. Chicago, IL University of Chicago Press. - References - Scientific Research Publishing, accessed December 10, 2025, <https://www.scirp.org/reference/referencespapers?referenceid=1348764>
33. Steps to an Ecology of Mind - Grokipedia, accessed December 10, 2025, <https://grokipedia.com/page/Steps_to_an_Ecology_of_Mind>
34. Context engineering: Best practices for an emerging discipline | Redis, accessed December 10, 2025, <https://redis.io/blog/context-engineering-best-practices-for-an-emerging-discipline/>
35. Citation - Thinking in systems : a primer - Search UW-Madison Libraries, accessed December 10, 2025, <https://search.library.wisc.edu/catalog/9910100084402121/cite>
36. Systems thinking - Wikipedia, accessed December 10, 2025, <https://en.wikipedia.org/wiki/Systems_thinking>
37. CYBERNETICS - The W. Ross Ashby Digital Archive, accessed December 10, 2025, <https://ashby.info/Ashby-Introduction-to-Cybernetics.pdf>
38. An introduction to cybernetics. by William Ross Ashby - Open Library, accessed December 10, 2025, <https://openlibrary.org/books/OL6202924M/An_introduction_to_cybernetics.>
39. Garfinkel, H. (1967). Studies in Ethnomethodology. Englewood Cliffs, NJ Prentice-Hall. - References - Scientific Research Publishing, accessed December 10, 2025, <https://www.scirp.org/reference/ReferencesPapers?ReferenceID=1808987>
40. Garfinkel, H - Studies in Ethnomethodology (1967) | PDF - Scribd, accessed December 10, 2025, <https://www.scribd.com/document/49927355/Garfinkel-H-Studies-in-Ethnomethodology-1967>
41. Vygotsky, L. S. (1978). Mind in Society The Development of Higher Psychological Processes. Cambridge, MA Harvard University Press. - References, accessed December 10, 2025, <https://www.scirp.org/reference/referencespapers?referenceid=2107373>
42. Leont'ev, A. (1978). Activity, Consciousness, and Personality. Englewood Cliffs, NJ Prentice-Hall. - References - Scientific Research Publishing, accessed December 10, 2025, <https://www.scirp.org/reference/referencespapers?referenceid=1831210>
43. LS Vygotsky.; Mind in Society : Development of Higher Psychological Processes, accessed December 10, 2025, <https://w.pauldowling.me/rtf/2021.1/readings/LSVygotsky_1978_MindinSocietyDevelopmentofHigherPsycholo.pdf>
44. Activity, consciousness, and personality - Semantic Scholar, accessed December 10, 2025, <https://www.semanticscholar.org/paper/Activity%2C-consciousness%2C-and-personality-Leont%E2%80%99ev-Hall/846dcce4ea934d56c452cc2ae6a118f916d8902b>
45. Quick question - how do you cite Aristotle and the Nicomachean ethics?? : r/askphilosophy, accessed December 10, 2025, <https://www.reddit.com/r/askphilosophy/comments/4ipxwq/quick_question_how_do_you_cite_aristotle_and_the/>
46. Aristotle: Nicomachean Ethics - ResearchGate, accessed December 10, 2025, <https://www.researchgate.net/publication/325449937_Aristotle_Nicomachean_Ethics>
47. Citation - Cicero on the ideal orator (De Oratore) - Search UW-Madison Libraries, accessed December 10, 2025, <https://search.library.wisc.edu/catalog/999908955302121/cite>
48. De Oratore - Wikipedia, accessed December 10, 2025, <https://en.wikipedia.org/wiki/De_Oratore>
49. Heidegger, M. (1962). Being and Time (J. Macquarrie, & E. Robinson, Trans.). Oxford, UK ‎& Cambridge, USA: Blackwell Publishers Ltd. (Original work published 1927) - ResearchGate, accessed December 10, 2025, <https://www.researchgate.net/publication/335060851_Heidegger_M_1962_Being_and_Time_J_Macquarrie_E_Robinson_Trans_Oxford_UK_Cambridge_USA_Blackwell_Publishers_Ltd_Original_work_published_1927>
50. Being and Time - Wikipedia, accessed December 10, 2025, <https://en.wikipedia.org/wiki/Being_and_Time>
51. Citation: Seeing like a state - BibGuru Guides, accessed December 10, 2025, <https://www.bibguru.com/b/how-to-cite-seeing-like-a-state/>
52. Seeing Like a State | Grantmakers in the Arts, accessed December 10, 2025, <https://www.giarts.org/article/seeing-state>