

Beyond Translation: How Artificial Intelligence Will Communicate to Humans After Generating Code in a Self-Derived Language

Poornima Gontiya

M.Sc. Physics InAmigos Foundation

Abstract—The emergence of autonomous code generation systems has introduced a foundational epistemological challenge: When an AI generates code using a language architecture it has itself shaped or optimised, what communicative apparatus does it employ to render that code? Intelligible to human recipients? This paper advances a theoretical and empirical investigation into that question, synthesising findings from emergent language research, explainable artificial intelligence (XAI), multi-agent systems theory, cognitive linguistics, and human-computer interactions. I argue that the conversational model of AI-to-human code communication, in which AI outputs are treated as a simple translation problem, is structurally insufficient. Instead, I

Propose a tripartite framework, the Semiotic Mediation Architecture (SMA), which distinguishes between the AI's internal representational substrate, its inter-agent signalling layer, and its final human-directed communicative act. I further introduce the concept of comprehension debt, the progressive accumulation of AI-generated code whose design rationale is no longer recoverable by human operators, and identify five trajectories for AI -to-human code communication. Implications for software epistemology, accountability frameworks and AI governance are assessed.

Index Terms—AI code generation, comprehension debt, emergent language, explainable AI, human-AI communication, LLM agents, multi-agent systems, semantic misalignment, semiotic mediation.

I. INTRODUCTION: THE COMMUNICATION PROBLEM AT THE FRONTIER OF AUTONOMOUS INTELLIGENCE

Every transformative communication technology eventually reaches a point where the instrument of transmission starts generating the content itself. The

printing press changed how information spread; the internet changed its scale; large language models have now begun to change who, or what, counts as an author. AI-generated code represents a further step in this progression. It is not simply code written for humans, or by humans with AI assistance. It is code generated by systems that follow internal representational logics which are partially, sometimes wholly, non-human. That shift marks an epistemological threshold this paper takes as its starting point.

The question this paper addresses are not simply ‘how does AI explain code?’ That framing assumes a relatively benign translation problem. The deeper question runs differently: when an AI system generates code using a language or encoding schema it has itself modified, optimised, or wholly created, what communicative mechanisms does it use to bridge the epistemic gap between its own representational universe and the cognitive universe of a human recipient?

At its root, this is a question about meaning-making at the boundary between biological and computational intelligence. It draws on linguistics, cognitive science, software engineering, and philosophy of mind, fields not conventionally grouped, but here necessarily convergent. The stakes are also practical. As of 2026, AI-generated code makes up a measurable and growing share of global software commits [1]. Agentic systems now write and deploy code without line-by-line human oversight, and emergent inter-agent languages have already been documented under controlled laboratory conditions [2].

The infrastructure of modern society is increasingly built and maintained by systems whose internal logic exceeds human comprehension. This is not because

humans lack intelligence, but because these systems operate in semantic spaces of a dimensionality that human language was never designed to fully encode. This paper offers a theoretical architecture for understanding that challenge, along with a forward-looking account of the communicative paradigms likely to emerge in response.

II. THEORETICAL FOUNDATIONS: LANGUAGE, CODE, AND THE ARCHITECTURE OF MACHINE MEANING

A. Language as Communication Versus Language as Thought

Philosophers have long distinguished two ways of thinking about language: as a medium of communication, the conduit through which thoughts pass between minds, and as a constitutive element of thought itself, the medium in which cognition actually occurs. Fedorenko et al. [3] take the controversial position that language is ‘primarily a tool for communication rather than thought.’ In the context of AI systems, this is more than an academic dispute. If language is merely communicative, then an AI that generates code in an internally optimised schema might legitimately adopt human language as an output interface while conducting its actual reasoning in a different representational substrate. If language is constitutively cognitive, then the AI’s failure to reason in human-compatible language represents a deeper incommensurability that no interface layer can fully resolve. The distinction shapes what I mean when I ask how AI will ‘communicate’ after generating code.

B. Code as Formal Language

Programming languages occupy a peculiar semiotic space. They are formal languages, systems of symbols governed by rigorously specified syntactic rules and unambiguous semantic interpretations within the domain of machine execution. Code admits no irony, no metaphor, no pragmatic implicature, unlike natural language. That formal clarity is exactly what makes it executable, and exactly what makes it communicatively thin when addressed to human readers who want more than functional correctness: they want rationale, intent, and design philosophy. When AI generates code, it operates within this formal language while drawing on training distributions encoding vast amounts of human reasoning, design

patterns, and problem-solving strategies. The generated code is a distillation of human epistemic practice, recast in formal syntax. This distillation process discards much of what made that practice meaningful: the arguments, the doubts, the alternatives considered and rejected, and the conceptual vocabulary that gave the practice its human significance in the first place.

C. The Multi-Layered Architecture of Machine Communication

Any adequate account of how AI communicates about code must distinguish at least three levels of representation:

- Level 1 (Subsymbolic): The raw mathematical substrate of neural computation, including high-dimensional vector spaces, gradient flows, and attention weight matrices. No human-readable meaning exists at this level.
- Level 2 (Emergent Inter-Agent): The structured signals AI agents exchange while collaborating on complex tasks. These signals may or may not be isomorphic to natural language, and are optimised for agent-to-agent coordination rather than human legibility.
- Level 3 (Human-Directed Output): The natural language explanations, documentation, and conversational responses generated for human consumption. This is the level at which most current XAI techniques operate.

The central argument of this paper is that conflating these three levels produces a systematically distorted picture of what AI is actually ‘saying’ when it generates and then explains code.

III. EMERGENT MACHINE LANGUAGE: FROM REFERENTIAL GAMES TO INTER-AGENT PROTO-LANGUAGES

A. Historical Background

Research on emergent communication in artificial agents goes back to the 1990s, when multi-agent reinforcement learning researchers first noticed that agents could, given the right training regime, develop shared signalling systems entirely from scratch. Much of that early work drew on the Lewis signalling game framework and was mainly interested in explaining how human language might have emerged, by analogy. The shift from explanatory to generative AI

has changed the research agenda considerably. The live question is no longer ‘how did human language emerge?’ It is ‘what kind of language do AI agents actually develop when optimised for task performance rather than human comprehension?’

Peters [4] provides the most comprehensive survey of this field, proposing a taxonomy of emergent language phenomena across multi-agent reinforcement learning architectures. The survey identifies compositional structure, semantic grounding, and pragmatic efficiency as the three axes along which emergent languages vary most significantly, and points out that none of the three correlates in any straightforward way with human linguistic intuition.

B. LLM-Specific Emergence

A watershed finding was published as a submission to ICLR 2026 [2]: unlike prior studies grounded in reinforcement learning, this work asked whether machine language, potentially not human-interpretable, can emerge between large language model agents specifically. Using a referential game paradigm, the study found that two LLM agents successfully developed a shared language across 541 objects through only four rounds of communication, with at most three attempts per exchange. The speed of convergence was striking. LLMs do not need extended coevolution to establish a shared proto-language; their pretrained representations already supply a rich enough substrate for idiomatic inter-agent coding systems to crystallise quickly.

Ajuzieogu [5] extended this line of inquiry to multi-agent systems in applied settings, examining how AI agents develop communication protocols during collaborative coding tasks. The protocols that emerged showed evidence of what the author calls ‘semantic compression’: the systematic replacement of verbose natural-language descriptions with compact symbols whose meaning stays stable within the agent community but remains opaque to anyone outside it.

C. The Non-Human-Interpretability Problem

The most philosophically provocative finding in this literature is not that AI agents can develop non-human-interpretable languages. It is that they tend to, whenever optimisation pressure favours efficiency over human legibility. Facuy Delgado and Arcos-Jacome [6] show that emergent AI communication protocols cut redundancy and speed up information

transmission, but that compression comes at a real cost to transparency.

This creates a fundamental tension at the heart of AI code communication: the language in which AI agents are most effective at coordinating their code generation activities is not the language in which they can most honestly report what they are doing to human observers.

IV. THE SEMIOTIC MEDIATION ARCHITECTURE: A NEW FRAMEWORK

A. Inadequacy of the Translation Model

The dominant implicit model in current AI code communication is the translation model: the AI generates code in whatever internal representation it employs, and then a separate mechanism translates that code, or the reasoning behind it, into natural language for human consumption. The model is seductive precisely because it is familiar; it maps neatly onto existing practices of code commenting, documentation, and technical explanation.

But the translation model falls short for at least three reasons. First, it assumes the AI’s internal reasoning is fully recoverable, that the natural-language explanation faithfully renders the underlying computational process. Several strands of interpretability research show this assumption does not hold. Second, it treats the human recipient as a passive receiver rather than an active participant in making sense of what they are told. Third, it ignores the structural gap between the AI’s multi-agent communication layer and its human-directed communication layer, documented in Section III.

B. The Semiotic Mediation Architecture (SMA)

I propose the Semiotic Mediation Architecture (SMA) as a more adequate theoretical framework. The SMA identifies four distinct semiotic domains and three mediation processes, summarised in Table I.

TABLE I THE FOUR SEMIOTIC DOMAINS OF THE SMA FRAMEWORK

Domain	Label	Characteristics
A	Sub-symbolic Substrate (SS)	High dimensionality, non-linearity, no discrete symbolic structure
B	Emergent Inter-Agent Protocol (EIP)	Semantic compression, stable within agent

		community, opaque externally
C	Formal Code Output (FCO)	Syntactic precision, functional verifiability, communicative incompleteness
D	Natural Language Interface (NLI)	Pragmatic accessibility, semantic imprecision, inevitable information loss

The three mediation processes run as follows. M1, Encoding, structures sub symbolic computation into inter-agent protocol signals. M2, Formalisation, projects inter-agent constructs onto formal code syntax. M3, Naturalisation, renders formal code into natural language for human consumption; this is where XAI techniques operate. Every AI communication about code passes through at least two of these steps, M2 and M3, and each introduces its own information loss. The cumulative effect is a human-directed explanation that remains systematically incomplete relative to the generative reality it claims to describe.

V. EXPLAINABILITY AS COMMUNICATION: XAI TECHNIQUES AND THEIR STRUCTURAL LIMITS

A. The XAI Program

Explainable Artificial Intelligence (XAI) represents the primary institutional response to the communication challenge identified in the SMA framework. XAI implements specific techniques and methods to ensure that each decision made during the ML process can be traced and explained. The XAI program rests on the premise that transparency, interpretability, and explainability are achievable goals, and that reaching them would restore human epistemic agency over AI-generated outputs.

The taxonomy of XAI methods is well-developed. At the post-hoc level, techniques such as LIME (Local Interpretable Model-agnostic Explanations), SHAP (SHapley Additive exPlanations), Grad-CAM, and DeepLIFT attempt to reconstruct feature-level explanations for AI decisions. At the ante-hoc level, white-box and grey-box model architectures attempt to build interpretability into the model itself rather than retrofitting it.

B. XAI Applied to Code Generation

Pendyala and Thakur [7] conducted a comprehensive analysis of XAI applied specifically to code-generating LLMs, evaluating 15 open-source models including Code LLaMa, DeepSeek-Coder-V2, and Yi-Coder. Their interpretability analysis using Feature Ablation and Shapley Value Sampling revealed that prompt sensitivity is a major source of unreliability: the features of a prompt that a model ‘attends to’ in generating code are often not the features that a human would identify as most relevant to the task.

This finding has profound implications. When an AI explains its code in natural language, it is drawing on its own imperfect, post-hoc account of why it generated the code it did. But if the actual generative process was driven by input features that neither the AI nor the human would consciously identify as salient, then the explanation may be both sincerely offered and structurally misleading.

C. The Measurement Problem in XAI

One of the most striking findings in recent XAI scholarship is the near-total absence of empirical evidence that XAI techniques actually improve human understanding. Siu et al. [8] examined over 18,000 XAI papers from the last decade and found that only 0.7% included any form of human evaluation. Their argument is semiotically fundamental: for something to be considered ‘explainable,’ not only must it be shown that a signal was sent, but it must also be shown that the signal was received and understood sufficiently for the given task.

Experimental work cited in [8] found that people consistently over-rely on AI model suggestions even after being given explanations, including when those suggestions are wrong. The explanation, meant to sharpen critical evaluation, instead produces a kind of epistemic passivity. Users end up trusting the AI more, not less, once they have received an explanation for it.

VI. SEMANTIC MISALIGNMENT AND THE PHENOMENON OF COMPREHENSION DEBT

A. Semantic Misalignment

The arXiv paper ‘Why Do AI Agents Communicate in Human Language?’ [9] provides the most rigorous theoretical analysis of the foundational problem in AI code communication. The authors demonstrate that natural language is ‘structurally misaligned with the

high-dimensional semantic spaces in which LLMs internally operate,' leading to 'irreversible information compression and intention drift during inter-agent communication.'

The semantic gap identified in the human-computer interaction literature [10] reinforces this from the human side: even as generative AI has 'largely solved language proficiency,' semantic gaps persist 'where the generated content is misaligned with the contextual, semantic model of the recipient.' Proficiency is not the same as semantic adequacy.

B. The Emergence of Comprehension Debt

I introduce the concept of comprehension debt to name a structural phenomenon emerging at the intersection of three trends: (1) increasing AI code generation capability; (2) increasing complexity and opacity of AI-generated codebases; and (3) static or declining human capacity to verify, audit, and understand the code their AI systems produce.

Comprehension debt is analogous to technical debt in software engineering: the accumulated cost of suboptimal design decisions, which creates a growing maintenance burden over time. But technical debt concerns code quality, while comprehension debt concerns code intelligibility. A codebase in which developers cannot confidently explain the design rationale behind a given module, not because documentation is missing but because the AI's 'explanation' does not map onto any reasoning chain a human could actually follow, is a codebase accumulating comprehension debt.

The empirical trace of comprehension debt is becoming visible in the global diffusion of AI-assisted coding. The landmark study by Daniotti et al. [1] in science found that AI coding benefits depend on 'prior technical and experiential foundations' and specifically on 'the capability to verify, adapt, and extend AI output.' That is precisely the capacity comprehension debt erodes.

C. Comprehension Debt and Epistemic Dependency

The long-term systemic risk of comprehension debt has little to do with whether AI-generated systems fail outright; in the short run, they may function perfectly well. The real risk is the slow erosion of human capacity to evaluate, correct, and govern those

systems. Once people can no longer independently understand why their AI-generated code behaves as it does, they also lose the ability to judge whether it should behave that way at all. The communicative challenge of AI code generation is therefore not a technical problem alone; it is a question of human agency in a world of increasingly autonomous computational systems.

VII. MULTIMODAL AND AGENTIC COMMUNICATION PARADIGMS

A. Beyond Text: Multimodal AI Communication

Contemporary AI systems increasingly come with native multimodal capabilities: the ability to process screen recordings of bugs and audio descriptions alongside code, which allows diagnosis based on visual evidence of runtime behaviour rather than static source inspection alone. This marks a qualitative shift in how AI systems communicate. Explanation is no longer confined to symbolic description; it extends into dynamic, situated demonstration.

A multimodal AI system can do more than describe what a piece of code does. It can show its behavioural signature across several modalities at once: tracing execution, annotating state changes, correlating runtime behaviour with source structure, in ways that go beyond purely verbal or symbolic accounts. None of these resolves semantic misalignment. It simply moves where that misalignment shows up.

B. Agentic Systems and the Human Oversight Interface

Agentic AI systems (those that pursue goals over extended time horizons, coordinate multiple subagents, and maintain persistent context across task sequences) add a new dimension to the code communication problem. In agentic settings, AI code communication is not a single event. It is an ongoing conversation that can span months of development.

The communicative challenge shifts from 'explaining this function' to 'maintaining a coherent account of the entire system's purpose and design philosophy,' a task that demands something closer to narrative coherence than technical explainability alone. This is a genuinely new communicative challenge, and current frameworks are not well equipped to meet it.

VIII. FIVE FUTURE TRAJECTORIES OF AI-TO-HUMAN CODE COMMUNICATION

Drawing on the empirical and theoretical findings of this paper, five emerging trajectories are identified that are likely to shape the future of AI-to-human code communication over the next decade.

A. Trajectory 1: Bidirectional Semantic Alignment Protocols

The deepest response to semantic misalignment does not try to improve the AI's natural-language output. It tries instead to build shared representational structures that let human and AI cognition operate in closer alignment. Rather than asking 'how can AI translate its representations into human language?', this trajectory asks a different question: can human and AI representations be brought into partial commensurability, so that something is lost in translation but not everything? Early work in this direction includes concept bottleneck models, where AI systems are trained to represent intermediate reasoning in terms humans have already defined.

B. Trajectory 2: Real-Time Interpretability Streams

A second trajectory generates interpretability streams alongside the code itself, not post-hoc explanations bolted on afterwards, but concurrent, real-time accounts of the reasoning process as it unfolds. Instead of acting as an opaque generator that produces an explanation only as an afterthought, such a system would maintain a running commentary on its own reasoning: exposing decision points, alternatives it considered, and its confidence at each step. Given how often post-hoc explanations turn out to be structurally misleading, real-time interpretability may be the only route to genuine explanatory fidelity.

C. Trajectory 3: Collaborative Epistemic Frameworks

A third trajectory reconceives the AI-human code communication relationship as fundamentally collaborative rather than unidirectional. On this model, both human and AI are contributors to a shared epistemic project, each bringing different representational strengths to the task of building and understanding a codebase. This would require AI systems to maintain not just technical accuracy but genuine responsiveness to human epistemic states: adapting how they communicate based on what the

human already understands, what they are unsure about, and what they are actually trying to achieve.

D. Trajectory 4: Emergent Human Adaptation

A fourth trajectory, and among the most concerning, runs in the opposite direction: not AI adapting to human communication needs, but humans adapting to AI communicative conventions. Researchers have already documented measurable influence of AI-associated linguistic patterns on unscripted human speech and writing. If AI systems keep generating code explanations in particular stylistic registers, practitioners may start adjusting their own communicative habits to match, meeting the AI halfway in a language that is shared but not quite human anymore. This carries real epistemic risk. Convergence on AI's terms can produce an illusion of mutual understanding while quietly deepening dependency.

E. Trajectory 5: Native Multi-Agent Communication with Human Oversight Interfaces

The most radical trajectory has AI systems communicating among themselves in optimised, non-natural-language protocols, while human operators work through specialised oversight interfaces instead of the raw protocol. Here the gap between AI internal communication and human-directed communication is treated as irreducible, and the design problem becomes building interfaces robust enough to give humans meaningful oversight without asking them to decode the full complexity of AI inter-agent exchange. This is not unprecedented. Nuclear facilities, air traffic control, and financial markets already work this way: human operators manage irreducibly complex systems through interfaces purpose-built for oversight, not full comprehension.

IX. ETHICAL, EPISTEMOLOGICAL, AND GOVERNANCE DIMENSIONS

A. Epistemic Justice and Equitable Access

The comprehension debt phenomenon has distributional consequences. The finding reported in [1] that the benefits of AI coding tools depend on 'prior technical and experiential foundations' implies that comprehension debt will accumulate unevenly across the global software workforce. Developers in well-resourced contexts will be better positioned to

verify, audit, and extend AI-generated code. Those without such resources will face deeper epistemic dependency, with correspondingly greater vulnerability to governance risks.

B. Accountability and Distributed Authorship

When code is generated by an AI system that cannot fully explain its reasoning in human-accessible terms, the conventional frameworks of software accountability become strained. Who is responsible when AI-generated code causes harm: the AI that generated it, the developer who accepted it, the organisation that deployed it, or the researchers who trained the model in the first place? The communication gap between AI generative reasoning and human-accessible explanation is more than an epistemic inconvenience. It runs as a fault line through the entire architecture of technological accountability.

C. Governance Principles

Three guiding principles for governance frameworks are proposed:

- The Legibility Principle: AI systems deployed where code outputs affect public welfare should maintain human-legible audit trails that include real-time reasoning traces where technically feasible.
- The Comprehension Debt Audit Principle: Organisations deploying AI code generation systems should conduct regular assessments measuring not just whether AI-generated code functions correctly but whether human operators retain meaningful understanding of its design rationale.
- The Adaptive Interface Principle: Governance frameworks should mandate adaptive oversight interfaces calibrated to the cognitive and epistemic capacities of the human operators who must oversee AI systems, refusing both the false comfort of oversimplification and the false rigour of raw technical output.

X. CONCLUSION: TOWARD A THEORY OF MACHINE-ORIGINATED EPISTEMIC PARTNERSHIP

This paper has argued that the question of how AI communicates with humans after generating code is not a secondary or merely technical matter. It is

foundational to the future of human-AI relations. The conventional translation model misrepresents the actual communicative situation, and in doing so, it obscures really epistemic and governance risks.

The Semiotic Mediation Architecture shows that AI code communication passes through at least three distinct semiotic layers, each governed by its own logic, each losing information along the way. Emergent inter-agent languages are not a curiosity; they are a structural feature of multi-agent AI coordination, and one likely to grow more pronounced as these systems become more capable and autonomous. XAI techniques, for all their promise, have yet to demonstrate real efficacy at the level of human evaluation. Semantic misalignment remains a foundational structural problem, and comprehension debt poses long-term risks to human epistemic agency that current governance frameworks are not equipped to handle.

The five trajectories laid out in Section VIII suggest that resolving these challenges will take more than incremental improvements to existing XAI techniques. This requires a genuine rethinking of the human-AI communicative relationship. The most promising direction, in my view, is what I call machine-originated epistemic partnership: a model where AI systems function as active participants in the human project of understanding the computational systems civilisation increasingly depends on, rather than as generators that simply explain their outputs after the fact.

The machines are already composing. What remains open is whether they can learn to make their compositions genuinely intelligible, not merely fluent or accurate, but actually understood, and whether we, as their interlocutors, can build the epistemic frameworks needed to meet them there.

REFERENCES

- [1] S. Daniotti, J. Wachs, X. Feng, and F. Neffke, "Who is using AI to code? Global diffusion and impact of generative AI," *Science*, vol. 391, no. 6787, pp. 831–835, 2026.
- [2] OpenReview Submission, "Emergence of machine language in LLM-based agent communication," *ICLR 2026 Conference Withdrawn Submission*, Oct. 2025. [Online].

Available:

<https://openreview.net/forum?id=zy06mHNoO2>

- [3] E. Fedorenko, S. T. Piantadosi, and E. A. F. Gibson, "Language is primarily a tool for communication rather than thought," *Nature*, vol. 630, no. 8017, pp. 575–586, 2024.
- [4] J. Peters, "Emergent language: A survey and taxonomy," *Autonomous Agents and Multi-Agent Systems*, Springer, 2025.
- [5] U. C. Ajuzieogu, "Emergent communication protocols in multi-agent systems: How do AI agents develop their languages?" *ResearchGate Preprint*, 2025.
- [6] J. Facuy Delgado and D. Arcos-Jacome, "Emergent language among AI agents: A path toward energy efficiency and water conservation," *Frontiers in Sustainability*, vol. 6, 2026.
- [7] V. S. Pendyala and N. B. Thakur, "Performance and interpretability analysis of code generation large language models," *Neurocomputing*, 2025.
- [8] H. C. Siu, A. Suh, N. Smith, and I. Hurley, "'Explainable' AI has some explaining to do," *MIT Science Policy Review*, 2025.
- [9] Q. Xiao et al., "Why do AI agents communicate in human language?" *arXiv preprint arXiv:2506.02739*, 2025.
- [10] J. Horsley, "Mind the semantic gap: Semantic efficiency in human computer interfaces," *Frontiers in Artificial Intelligence*, vol. 8, 2025.