KEY IDEAS SUMMARY

ULOGIC: A Universal (Mathematical-Computational) Language for Verifiable Neuro-symbolic Reasoning

© Leopoldo Cano Guardiola 2025 - https://ulogiclang.ai

Introduction

Artificial intelligence and the foundations of mathematics are facing parallel crises. On one hand, Large Language Models (LLMs), despite their impressive capabilities, suffer from a fundamental reliability problem, manifested in "hallucinations" and an inability for rigorous reasoning. On the other hand, the foundations of mathematics, although functional, still rely on axiomatic systems like ZFC, which are designed to circumvent paradoxes rather than resolve them at their root (a ZFC system that also suffers from insufficiency—if it has a model, it has a countable one, meaning the axioms do not sufficiently characterize the intuitive idea of a set).

This report introduces (without delving into its definition and details, only explaining its capabilities) ULOGIC, a formal language and system that proposes a unified solution to both crises. ULOGIC represents a paradigm shift in formal systems by unifying syntax, semantics, and computation into a single expression-based framework. Its procedural, self-referential, and constructive nature internally resolves foundational paradoxes and, in doing so, lays the groundwork for a novel neuro-symbolic architecture. This architecture is capable of producing AI-generated reasoning that is verifiable and reliable, culminating in the vision of a global network of reusable and accurate knowledge known as TekDocs.

This document explores this proposal in three parts. Part I will establish the formal foundations of ULOGIC, detailing its basic principles and contrasting them with established logical traditions. Part II will detail its application in a neuro-symbolic AI system, demonstrating how it overcomes the limitations of current LLMs. Finally, Part III will explore the transformative vision of TekDocs (Transportable Encapsulated Knowledge Documents) as a global substrate for cumulative and verifiable knowledge.

Part I: The Foundational Architecture of ULOGIC

This section shows how ULOGIC is a novel formal system, summarizing its basic principles and contrasting them with established logical traditions.

Section 1. A Unified Framework of Procedural Expressions

The central principle of ULOGIC is that "all constructs are expressions." In this system, there is no fundamental distinction between data, definitions, theorems, proofs, algorithms, or even the execution of those algorithms. They are all simply expressions with an internal structure and recognizable parts, independent of their serialized representation (their written form).

ULOGIC operates on a set of **rules** that explain how to obtain some expressions from others. Everything is fundamentally a "procedure." These procedures can be of two types:

- **Open**: The operational application of rules, which is the essence of reasoning.
- **Closed**: The application of rules following an algorithm to generate sequences of contexts on which reasoning is performed.

Fundamentally, reasoning and computation are the same procedural activity within the system.

ULOGIC is not based on type theory, an explicit departure from systems like Coq or Agda, which rely on types to maintain consistency. The validity of an expression is a matter of its **internal structure** alone, verifiable by analyzing its conformity with the system's rules.

Section 2. Internal Semantics and the Radical Rejection of External Models. Need for a complete re-evaluation and re-foundation of modern logic (Tarskian semantics, Gödel's theorems, formal logic...)

In ULOGIC, "there is no external semantics in models; meaning is internal without reference to anything external." This principle represents a fundamental break with the standard approach to semantics in logic: Alfred Tarski's model theory, where meaning is a correspondence with an external mathematical structure. ULOGIC holds that all of Tarskian semantics is not only unnecessary but a complete conceptual mistake that has led current formal languages, like first-order logic (FOL), to a dead end.

In ULOGIC, by contrast, the "meaning" of an expression is its structural and procedural relationship with all other expressions within the system. Meaning is implicitly defined by the rules that can produce it and the rules in which it can participate to produce new expressions. It is a purely syntactic, operational, or proof-theoretic semantics. The system is self-contained and self-justifying.

This approach is reminiscent of the philosophical school of formalism, where mathematics is considered the manipulation of symbols according to specified rules, without concern for their "meaning" in an external reality. But it differs radically from formalism in that, on one hand, it moves away from the "alphabetic" view, because expressions ARE NOT sequences of letters from an alphabet: Written expressions are serializations of a structure (the expression) that can be "written" in any way following any convention. And on the other hand, ULOGIC moves away from the typical and characteristic formalist assertion of "manipulation of symbols without meaning" because it is absolutely—and ironically—erroneous: **it is** the manipulation of symbols **using** rules that **creates** the meaning *(*intensional meaning, no denotational)*

To put it plainly: The main problem with Tarskian semantics, which interprets a formal language in a "mathematical structure" (a set with functions and relations), is that it is NOT a semantics. It seems plausible because it emulates denotational semantics, where signs refer to external realities captured by a perceptual system. But denotational semantics is radically different from the interpretative semantics of Tarski and standard symbolic logic.

To understand it better: Formal logic, semantics and model theory, and all of "modern" logic began as an attempt to build a language, and has ended up becoming just another mathematical theory, far from the original foundational attempt: It is a theory of POLYNOMIALS, simply.

What is a polynomial? Given a structure on a set A, with functions f1, f2... and relations R1, R2... we can form sequences that we call terms and formulas. If the functions are simply an "addition" (+) and a "multiplication" (·), the "polynomials" are just the usual polynomials like $3x^2+x^3+2$ (which, by the way, Descartes invented without knowing it).

"Tarskian semantics" is merely an EVALUATION of polynomials, that is, a mathematical interrelation between two structures. Evaluating a polynomial from a multiplicative ring is the famous operation we learn in school of "substituting the x." In formal logic, "terms" are polynomials whose evaluation is a value in the base set A, and "formulas" are ALSO polynomials whose evaluation actually takes a value in the set {0,1}, which we call "true-false" but which has NOTHING to do with the usual meaning of "true and false."

The polynomial evaluation function allows for the definition of the well-known property "polynomials that are satisfied in a structure." And from there, to define a "deduction function" as follows: from the polynomials {p1, p2, ...pn}, the polynomial q is "deduced or derived" if and only if it occurs that q "is satisfied" in every structure A where p1, p2...pn are satisfied.

This derivation function is the only one really used in model theory and mathematical logic. That is why "syntactic deductive calculi" (sets of rules for deriving polynomials from each other) are irrelevant: they can be Hilbert-style axiomatic, Gentzen-style natural-sequential, or any other combination, because it doesn't matter, as they are constructions that end up with the SAME derivation function, the one defined using polynomials through Tarski's interpretation method.

Polynomials, in turn, can be used to define "sets of structures." Given a set of polynomials P, Struct(P) is the set of structures that "satisfy" all the polynomials in P (that is, all polynomials in set P have the value "1" when evaluated in any structure A from the set Struct(P)).

<u>Question 1:</u> Are the structures in the set Struct(P) isomorphic to each other? The answer is no. The Löwenheim-Skolem theorems say just that. In other words, trying to "define" a structure "using polynomials" is a very bad idea.

<u>Question 2:</u> Are the structures in the set Struct(P) "elementarily equivalent to each other"? (Two structures are "elementarily equivalent" if any polynomial p that is satisfied in one is also satisfied in the other). Well, the answer is almost never, even for very simple cases like trying to define the structure of the natural numbers. This is the only meaning of the famous Gödel's incompleteness theorem (which has nothing to do with "unprovable-truths").

Therefore, if we use "polynomials" and the convoluted way of "defining sets of structures" through "Tarskian semantics" (which is merely a polynomial evaluation), the result is of little use: we achieve neither isomorphism nor even elementary equivalence.

These results, including Gödel's, are IRRELEVANT to mathematics and to logic (if we understand "logic" as an attempt at reverse-engineering the real mathematical language), because no mathematician in history has defined mathematical structures using "polynomials" (polynomials are interesting mathematical objects, but nothing more). If you define mathematical structures (like that of the natural numbers) or any other "using normal mathematical expressions," you of course achieve isomorphism and sufficient accuracy.

Formal logic is a trap of terms and definitions: "Formulas" are not mathematical expressions (although they initially wanted to be), as they have ended up being merely polynomials in structures. Tarskian "truth" is not what it claims to be, and it has nothing to do with the word "truth" in its usual uses; it is merely a polynomial evaluation. "Deduction" is not deduction in the mathematical sense of "real mathematics"; it is merely an interrelation between polynomials based on the Tarskian evaluation function.

Gödel's incompleteness, which using formal logic terminology is stated as "there are unprovable truths" (correct terminology in model theory), seems like a miracle, but that is because the meaning of "truth" and "provable" are falsified in the terminology of model theory (Gödel's theorems simply state the inability to define sets of elementarily equivalent structures using the convoluted method of Tarskian satisfaction).

Formal logic began as an attempt to continue the venerable tradition of "reverse-engineering" by Aristotle, Peano, Frege, Russell, etc. (that is, the attempt to find the rules we follow when we do proofs, rules that, once "discovered," transform and reinvent the very language they are studying). But the technical skill and genius of the mathematicians who tried to go further (Tarski, Hilbert, Gödel, and so many others) have led us to a dead end: to a mathematical theory of "polynomials" that has abandoned the aspirations of understanding real mathematics.

Real mathematics, the real deal, is not "what is inside the polynomials" but what is outside in the so-called metamathematical proofs.

This, in turn, leads us to the biggest mistake in history, initiated by brilliant minds like Hilbert: To conceive that there are "alphabetic formal languages" on one hand, with exact rules, and on the other an "informal-metamathematics" that is not formalizable, but which uses indubitable-finitistic reasoning methods to make proofs about formal systems.

The manifest falsehood of such a research program is even obvious: Just go to a mathematics faculty and see WHAT mathematicians do. And what they do is handle a language (quite complex) that is the heritage of a historical process, a language within natural language, a language that is now plagued with previously nonexistent symbols and expressions—created by logical research.

But it is a single language. A language in which we reason, prove, define, define algorithms, execute algorithms, and even talk about the language itself. Does the reader know anyone on planet Earth who has "stepped out" of language to talk about language? We only have ONE single language.

Section 2b. We need to redefine "logic" as an absolutely necessary reverse-engineering project to understand the ONLY language we have (a language with mathematical, computational, and metalinguistic capabilities)

The word "logic" has meant so many things in the last 200 years (the famous Russellian logicism even has a mystical point) that it barely means anything anymore. Our proposal is very simple and even self-evident, and is summarized in three principles:

1. **A single language with multiple capabilities** (discursive, mathematical, computational, and metalinguistic): Human beings have a single language. We have created this language over 3000 years. "Natural language" is the backdrop, but within it, we have created sub-languages like those of mathematics, computation, and algorithms. But it is all a single language: one capable of making proofs, algorithms, executions, and metalinguistics.

2. **"Logic" as "reverse-engineering"**: "Logic" must change course: it must be the "reverse-engineering" project to decipher what rules we follow when we make proofs, algorithms, computations, or metalinguistics. The current language is blurry and imprecise. When we find "patterns" and turn them "into rules," we are transforming the language into something new! It is the iterative process we have followed in creating mathematics: new people create new methods without knowing it, and people who come after them study them and create rules that change the previous language. Today's "evident logical rules," which a mathematician knows, are part of the known historical process. But the important question is always: What are the logical rules we are using right now when doing mathematics, computation, and metalinguistics, rules that are there and of which we are not even aware? The answer is surprising: almost everything that matters and is essential is currently "off the radar" of logic.

3. **Progress consists of creating more powerful languages**. The progress of civilization is based exclusively on this evolutionary process of creating an increasingly powerful language along with complex ways of "applying" it to reality to learn to see what is not visible: It is the story that begins before Euclid (and which Euclid crystallized into a new form of reasoning) up to Newton, and ending in nuclear energy, space exploration, and genetic engineering. The ultimate superpower of the human species is language, something that began as simple statements to describe reality, and has ended up as an imposing edifice of "reasoning" and "computation."

Summarizing and concluding:

By rejecting external models, ULOGIC defines a completely self-contained logical-computational universe. The "truth" of a statement is equivalent to its "provability" or "constructibility" within the system.

This has profound implications, and, put this way, merges the concepts of truth and computation. Although it would be best to stop talking about "truth" and reserve the term "truth" exclusively for the denotational semantics of the interrelation of language with the real world (but that is a project outside of logic and mathematics, because it involves real perceptual systems that bridge perceptions and language through mappings in internal perceptual topological spaces where certain regions of that space are dynamically labeled using language... and yes, it sounds like what it is: that's another story).

Gödel's incompleteness theorems demonstrated a gap between "truth" and provability in formal systems; there are statements that are "true" (in the standard model of arithmetic) but not provable within the system. But in reality, all of that is merely a way of speaking in the universe of Tarskian semantics and polynomial theory.

We could say that ULOGIC closes this gap by **defining** truth as provability. But we can also say that ULOGIC dissolves this problem, which ceases to be a problem. A statement for which a constructive proof-expression cannot be formed is not "true but unprovable"; it is simply not a theorem of the system.

This eludes the philosophical difficulties of Gödel's theorems by refusing to recognize a notion of truth external to the system itself, achieving in return complete internal coherence where truth and proof are synonymous. Although we insist: the exact meaning of Gödel's theorems is simply what was explained above (that using polynomials to define sets of structures through Tarski's satisfaction method is a bad idea that does not even manage to achieve elementary equivalence between structures... and of course, it is also a method that no mathematician has ever used).

Section 3. The Only Possible Solution to Set-Theoretic Paradoxes: Definitions as Non-Eliminable Constructs

Expressed in usual terms according to the literature, naive set theory is based on an unrestricted comprehension principle that leads directly to Russell's Paradox. Standard solutions, like ZFC or type theory, impose axiomatic or structural restrictions to avoid the formation of paradoxical sets.

ULOGIC proposes a deeper and more radical solution, based on a key idea: **mathematical definitions ARE NOT eliminable abbreviations**, contrary to what the logical standard dictates.
In traditional logic, a definition is a mere notational convenience that can be replaced by its expanded form without altering the meaning. ULOGIC rejects this notion. Here, definitions are expressions with their own internal structure and a fundamental role in the system.

"Definitions that are not abbreviations and are not eliminable" have always existed. But it is in the 19th century that sets begin to be seen as objects of study (is the set of continuous functions closed within the set of bounded functions under a certain metric?). Cantorian theory accelerated this process. And here a leap and a train wreck occurred: the set-theoretic contradictions.

It is precisely the "discovery" of this non-eliminable nature of definitions that allows ULOGIC to resolve the set-theoretic paradoxes at their root. In ULOGIC, the rules for creating definition expressions are designed to prohibit the construction of this type of structurally defective recursion. A definition is not a simple abbreviation, but an expression that did not exist before and is now introduced into the system.

An expression "that did not exist before, but is now introduced into the system" is exactly what one understands by "axiom" or "hypothesis"... and that, in essence, is what definitions are: An authorization to create new expressions that did not exist before.

"Defining" is not making abbreviations (although it can be): In reality, "defining" (when definitions are not eliminable) is having permission to create "axioms" (expressions that did not exist before) but without calling them axioms. Then we can prove things that would be impossible to prove without those "axioms" and then erase the tracks and pretend that what we have proven only with the help of definitions would also have been provable without the help of definitions (which is completely false).

ULOGIC explains all this with millimeter precision.

But if the reader wants to see "the light at the end of the tunnel," we propose an exercise: The famous Cantor's theorem that the cardinality of |A| is less than that of its power set |P(A)| is well known, and is based on a usual technique: making a magical definition that proves the theorem by reductio ad absurdum. Could the reader provide a proof of that theorem WITHOUT making the key intermediate definition? It is impossible.

Now the question: If definitions are eliminable abbreviations that add nothing new to what already existed, why is there a mountain of theorems that CANNOT be proven without making a magical intermediate definition?

This pattern is well known but poorly understood ("becoming a professional mathematician" consists of intuitively mastering this method). In fact, and as a very significant example, the "mathematical reality" that there are infinities of different cardinalities and the various proofs thereof seem like an amazing discovery of a transcendent reality, but they are based on a proof pattern that is fundamentally (if you will) "a grammatical rule," a mere convention "of how to use expressions," a convention that we do not even know we are following. (This can only be "seen" if you have explicit rules and alternative languages like ULOGIC).

Indeed: we are slaves to our language and cannot see beyond what the language allows us. Only new languages allow us to see the naiveties, inaccuracies, and defects of old languages (interesting: research the history of mathematics books that talk about the evolution of mathematical rigor, because it has been precisely that process of language evolution).

Continuing: The rules of the ULOGIC system prevent a definition expression from being constructed in a way that generates a vicious circular dependency, making paradoxes syntactically impossible to formulate. This is analogous to a syntax error in a programming language, but applied to the very structure of reasoning.

Table 1: A (Possible-Useful-but-Imperfect) Comparative Analysis of Paradox Resolution Mechanisms

SYSTEM	BASIC PRINCIPLE FOR SET/COLLECTION FORMATION	PARADOX AVOIDANCE MECHANISM	ALLOWS UNIVERSAL SET	KEY CONSEQUENCE
NAIVE SET THEORY	Unrestricted Comprehension	None (Inconsistent)	Yes (leads to paradox)	Inconsistent
RUSSELL'S TYPE THEORY	Comprehension by types	Syntactic type hierarchy	No	Rigid hierarchy
ZFC	Axiom Schema of Separation	Restriction to subsets of existing sets ("Limitation of Size")	No	Cumulative hierarchy (Axiom of Foundation)
NEW FOUNDATIONS (NF)	Stratified Comprehension	Syntactic stratification of formulas ("Limitation of Structure")	Yes	Non-well-founded models possible / AC fails
ULOGIC	Rules for the formation of non-eliminable definition expressions	Procedural rules preventing definitions with vicious circular dependencies	Yes	Definitions are fundamental constructs / Consistency by construction

Section 4. The ULOGIC Language as its Own Metalanguage. Is it possible?

That an "exact-formal language" can be its own metalanguage is today a logical heresy, and anyone who claims it will seem "a bit foolish" because it is well-proven that this is impossible.

However, the number of theorems in mathematical logic that *are not what they seem* is endless.

Tarski's indefinability theorem proves that a formal language cannot define its own semantic truth predicate without falling into contradiction. The standard solution is Tarski's hierarchy of languages, which separates the *object language* from the *metalanguage*.

ULOGIC makes the radical claim that "the language has the capacity to be its own metalanguage." This is possible because ULOGIC eludes Tarski's theorem. Having abandoned Tarskian semantics and external models, ULOGIC does not operate with a concept of semantic truth, but with the syntactic and procedural concept of "being a validly constructed expression" or "being a theorem".

ULOGIC's capacity for safe self-reference derives directly from its foundational principles: expressions are structures, definitions are not eliminable, and semantics is internal.

A ULOGIC expression can refer to another expression and assert its *provability* or *constructibility*. The Liar Paradox ("This sentence is false") is deactivated because the statement "This expression is not a theorem" is not inherently paradoxical. The internal consistency of ULOGIC is maintained by its definition rules, which prevent the construction of expressions that would lead to contradictions. This concept is analogous to computational reflection, where a program can inspect and modify its own code as if it were data, but elevated to a fundamental principle of logic.

The problem for ULOGIC in trying to create an "exact language with metalinguistic capacity" is "how to achieve it," because that is an unexplored path. (SNEAK PEEK: We will show that it is even possible for the language to create models of itself... although this is a novel field of research).

Part II: The Neuro-symbolic Symbiosis: ULOGIC and Large Language Models

ULOGIC is a pure research project in the foundations of logic and mathematics, with philosophical implications, and in principle destined to exist within books of philosophy, logic, or in meetings of strange people (logicians, mathematicians, and philosophers).

This is not very encouraging, because the applied mathematics community considers "logic" to be irrelevant to them (and they are not wrong, given the evolution of logic which has ended up as polynomial theory). Pure logicians working on things like advanced model theory or set theory in a universe of forcing techniques to prove the independence of "axioms" only talk among themselves. And philosophers of logic and mathematics analyze the landscape, reaching inevitable conclusions: today, nobody has the faintest idea what mathematics or logic is (and the accumulation of paradoxes and problems has simply been resolved "by forgetting": forgetting about a problem is another way to "solve" the problem in a community).

In other words: ULOGIC has not the slightest chance of having an impact in the real world if the proposal is made to mathematicians-logicians (perhaps philosophers of science have a broader perspective and will welcome it more).

However, as so often happens, it could have an unforeseen impact: In Artificial Intelligence.

Artificial intelligence research in recent years (2016-2025) has produced an astonishing explosion of generative "prodigious machines." But it has stagnated: no possible path is seen to improve the logical and reasoning capabilities of systems in a "powerful" way and not merely "simulated."

As so often in history, the solution could lie in something that originally has nothing to do with Artificial Intelligence, the result of research that never aimed to be applied to artificial intelligence.

Now we are going to tell that story.

Section 5. Uniting Intuitive and Formal Reasoning

The cognitive framework "Thinking, Fast and Slow" describes two modes of thought: System 1, which is fast, intuitive, and pattern-based, and System 2, which is slow, deliberate, and logical. This paradigm can be mapped directly to AI architectures.

- **LLMs** are powerful System 1 engines, excelling at pattern recognition, fluency, and intuitive leaps.
- **Symbolic AI**, on the other hand, represents pure System 2 engines, excelling at rigorous, verifiable, step-by-step reasoning.

The key weakness of LLMs—their propensity for hallucinations, lack of genuine reasoning, and poor reliability—stems from their lack of a System 2 component.

The goal of neuro-symbolic AI is to create a hybrid architecture that combines the strengths of both approaches, uniting perception and intuition with logic and verification.

But if on the "symbolic" side (System-2) we use "toy" languages like first-order logic and so many other current ones, the result is going to be terribly discouraging: It will be like trying to write Don Quixote using two vowels and five consonants... pieces are missing!

Section 6. The ULOGIC Verifier Architecture

The proposed neuro-symbolic architecture around ULOGIC functions as an iterative loop of generation and verification:

1. **Input (Prompt)**: A user poses a problem that requires a proof, an algorithm, or a formal procedure.
2. **LLM (Generator)**: The LLM, acting as a System 1 "intuition" engine, generates a candidate solution in the form of a ULOGIC expression. It is not expected to be perfect; it is a hypothesis.
3. **ULOGIC (Verifier)**: The ULOGIC engine, a deterministic System 2 reasoner, receives the expression. It does not interpret its "meaning" but performs a purely syntactic check to determine if the expression is well-formed according to its rules. This process is a form of **formal verification**.
4. **Feedback**:
   - **Success**: If the expression is valid, it is accepted as a correct proof,

algorithm, or procedure.
    - **Failure**: If the expression is invalid, the ULOGIC verifier returns a precise structural error report, pointing out the exact sub-expression and the rule that was violated.

5. **Refinement:** The LLM receives this structured feedback. This feedback becomes part of the context for its next attempt, guiding it to correct its error. This iterative process continues until a valid expression is generated.

This loop is a cognitive training process. Through thousands of these interactions, the LLM adjusts to generate valid ULOGIC expressions.

The LLM, by being trained on ULOGIC, a language with advanced capabilities and sufficient expressive power, **learns the intuition of how to make proofs, algorithms, and procedures**; it learns to think logically and to create verifiable, not merely statistical and imitative, reasoning.

The structured feedback from the verifier provides a powerful training signal that goes beyond simple next-token prediction, shaping the LLM's internal representation space to be compatible with the underlying logic.

Creating this architecture has its technological challenges (an orchestrator is needed to interconnect the two systems; on the LLM side, the content is encoded in the abstract vector embeddings used by transformer architectures, but which represent TekDocs, documents containing theorems, definitions, proofs, algorithms, algorithm executions—TekDocs that are human-readable expressions, and stored in relational and other types of databases). It sounds complex, and it is. But it is not the most complicated part.

The monumental challenge is first to specify the ULOGIC language (which a person understands because we know how to navigate swampy terrains) to a sufficient degree of precision to allow the construction of a Kernel-Verifier.

Section 7. From Imitative Statistics to Verifiable Cognition

The final product of this neuro-symbolic system is not the natural language text generated by the LLM, but the **verified ULOGIC expression**. This result is fundamentally different from traditional AI-generated content and possesses transformative properties:

- **Verifiable**: Its correctness can be mechanically and deterministically checked by any ULOGIC engine or Kernel.

- **Reliable**: It is guaranteed to be free of logical contradictions and hallucinations, as these would be detected by the verifier.
- **Explainable**: The ULOGIC expression itself **is** the explanation. A proof-expression is a complete and structural account of the reasoning.
- **Composable**: Being a formal expression, it can be reliably used as a component in larger and more complex constructions.

This approach directly addresses the main criticisms of current generative AI, producing reasoning that is not only fluent but also correct, transparent, and trustworthy.

Part III: TekDocs (Transportable Encapsulated Knowledge Document) A Global Substrate for Reusable and Accurate Knowledge

This final part extrapolates the consequences of the neuro-symbolic system, arguing for a revolution in knowledge management and collaboration.

Section 8. Transcending the Knowledge Representation Bottleneck

The goal of the Semantic Web and knowledge representation has long been to create machine-readable knowledge. The main tools for this purpose are ontologies and knowledge graphs.

However, this field has been hampered by the "knowledge acquisition bottleneck." The manual creation and maintenance of large, coherent ontologies is an extremely expensive, slow, and fragile process.

The ULOGIC-LLM system offers a solution to this problem by automating the creation of *formal and verifiabl***e** knowledge. It bypasses the manual bottleneck by having the AI do the heavy lifting of formalization, guided and corrected by the ULOGIC verifier.

Section 9. The TekDocs Network Architecture

A "TekDoc" (Transportable Encapsulated Knowledge Document) is defined as a verified and self-contained ULOGIC expression that represents a unit or set of exact knowledge (definitions, arguments, proofs, theorems, algorithms, algorithm executions, complete theories...).

The vision is a worldwide network of TekDocs that would contain the products of reasoning, algorithms, and definitions, creating a global ecosystem of reusable and accurate knowledge.

The properties of this network would be:

- **Reusability**: Any TekDoc can be referenced and incorporated into the creation of a new TekDoc, as they all share the same language.
- **Accuracy**: Every piece of knowledge in the network is formally verified. Ambiguity and contradiction are eliminated.
- **Accumulative Growth**: Science and engineering can progress in a truly cumulative way, building on a shared and solid foundation of verified results.

This TekDocs network can be understood as a "**Git for formal knowledge**."

Before Git, collaboration in software development was difficult. Git introduced a decentralized model where code is managed in self-contained and versioned units (commits) that can be easily forked, merged, and shared, which led to an explosion in collaborative open-source development.

Today, scientific and technical knowledge is largely stored in human-readable documents (PDFs), which are ambiguous, non-executable, and difficult to integrate. TekDocs would function as "commits" of formal knowledge: atomic, verifiable, and with a clear dependency graph.

The TekDocs network would therefore function like a **GitHub for science and engineering**. A physicist could "fork" a General Relativity TekDoc, add a new term, and submit a "pull request" that would be automatically checked for logical consistency. An engineer could "import" a TekDoc for a sorting algorithm, with the guarantee that it is correct, into the design of a larger system. This would drastically accelerate innovation and avoid the constant "reinvention of the wheel" and propagation of errors that are common today.

Conclusion: A Paradigm Shift in Formal Thought and Artificial Intelligence

The foundational innovations of ULOGIC—unified structural expressions, non-eliminable definitions, internal semantics, and safe self-reference—are not mere theoretical curiosities.

They are the necessary substrate for a new generation of neuro-symbolic AI.

This system, in turn, is not an end in itself, but a "machine for producing accuracy": the engine that will populate the TekDocs network, creating a shared, global, verifiable, and reusable common intellectual universe.

ULOGIC is positioned as a potential turning point in the history of logic and computation.

It offers a path towards truly intelligent and reliable AI, and at the same time lays the foundation for a more rigorous and collaborative future for human knowledge, simultaneously addressing the foundational crises that have limited both fields for decades.

Bibliographic Reference

Model Theory and Mathematical Logic

- Chang, C. C., & Keisler, H. J. (2012). Model Theory (Third Edition). Dover Publications.
  An absolute classic and an encyclopedic reference on model theory, reissued by Dover.
- Enderton, H. B. (2001). A Mathematical Introduction to Logic (Second Edition). Academic Press.
  Considered one of the best introductory texts to mathematical logic, covering the completeness and incompleteness theorems with great clarity.
- Hodges, W. (2008). A Shorter Model Theory. Cambridge University Press.
  A condensed version of his monumental "Model Theory," ideal for a graduate-

level introduction to the field.

- Marker, D. (2002). Model Theory: An Introduction. Springer.
A modern reference text in model theory, focused on the applications of the field in algebra and geometry.
- Tarski, A. (1983). Logic, Semantics, Metamathematics: Papers from 1923 to 1938. Hackett Publishing Company.
A collection of Alfred Tarski's foundational works where he develops his theories on truth and formal semantics.

Philosophy of Mathematics and Gödel's Theorems

- Goldstein, R. (2005). Incompleteness: The Proof and Paradox of Kurt Gödel. W. W. Norton & Company.
An acclaimed intellectual biography that accessibly explains the life of Gödel and the profound impact of his theorems.
- Horsten, L. (2011). The Tarskian Turn: Deflationism and Axiomatic Truth. The MIT Press.
A book on the philosophy of logic that deeply analyzes Tarski's theory of truth and its implications for modern debates.
- Linnebo, Ø. (2017). Philosophy of Mathematics. Princeton University Press.
An excellent and very current introduction to the main currents of the philosophy of mathematics (Platonism, structuralism, logicism) and their central problems.
- Shapiro, S. (2005). The Oxford Handbook of Philosophy of Mathematics and Logic. Oxford University Press.
An indispensable reference work, with essays by leading specialists on practically all relevant topics in the philosophy of logic and mathematics.
- Smith, P. (2013). An Introduction to Gödel's Theorems (Second Edition). Cambridge University Press.
Considered the best book for a technical but guided introduction to the proofs of the incompleteness theorems, ideal for students of philosophy, logic, and mathematics.

Articles on Neuro-symbolic Architecture (arXiv)

- d'Avila Garcez, A. S., & Lamb, L. C. (2020). Neurosymbolic AI: The 3rd Wave. arXiv:2012.05876.
  A perspective article that positions Neuro-symbolic AI as the "third wave" of AI, describing its principles and potential.
- Mao, J., Gan, C., Kohli, P., Tenenbaum, J. B., & Wu, J. (2019). The Neuro-Symbolic Concept Learner: Interpretable Reasoning, Compositional Concepts, and Robust Learning. arXiv:1904.12584.
  An influential work that presents a concrete architecture for concept learning and visual reasoning in an interpretable way.
- Sarker, M. M. K., et al. (2021). Neuro-Symbolic Artificial Intelligence (AI): A new-age of AI. arXiv:2105.05330.
  A review article that provides an overview of the motivations, challenges, and different approaches within the field of Neuro-symbolic AI.
- Yi, K., Wu, J., Gan, C., Torralba, A., Kohli, P., & Tenenbaum, J. B. (2018). Neural-Symbolic VQA: Disentangling Reasoning from Vision and Language Understanding. arXiv:1810.02338.
  Proposes an architecture for visual question-answering (VQA) systems that explicitly separates neural (perception) from symbolic (reasoning) modules.
- Enhancing Large Language Models through Neuro-Symbolic Integration and Ontological Reasoning - arXiv, https://arxiv.org/html/2504.07640v1
- Design Patterns for LLM-based Neuro-Symbolic Systems - Neurosymbolic Artificial Intelligence, https://neurosymbolic-ai-journal.com/system/files/nai-paper-787.pdf
- Accelerating Knowledge Graph and Ontology Engineering with Large Language Models, https://arxiv.org/html/2411.09601v1

Accessible and Interesting Introductory Content

- Formal proof - Wikipedia, https://en.wikipedia.org/wiki/Formal_proof
- Type theory - Wikipedia, https://en.wikipedia.org/wiki/Type_theory
- Formal system - Wikipedia, accessed: July 8, 2025, https://en.wikipedia.org/wiki/Formal_system
- New Foundations - Wikipedia, https://en.wikipedia.org/wiki/New_Foundations
- Self-reference - Wikipedia, https://en.wikipedia.org/wiki/Self-reference
- Neuro-symbolic AI - Wikipedia, https://en.wikipedia.org/wiki/Neuro-symbolic_AI

- Formal verification - Wikipedia, https://en.wikipedia.org/wiki/Formal_verification
- Logical form - Wikipedia, https://en.wikipedia.org/wiki/Logical_form
- Model theory - Wikipedia, https://en.wikipedia.org/wiki/Model_theory
- Proof-theoretic semantics - Wikipedia, https://en.wikipedia.org/wiki/Proof-theoretic_semantics