GRAPHS.SGIT.AI · SECOND EDITION OF THE ARGUMENT

Fractal Semantic Graphs

Meaning Through Connectivity

A node connected to nothing means nothing. What a thing is emerges from the edges traceable from it, and how much you can rely on that meaning is a function of how richly and how independently it is connected.

The first edition argued that from first principles. Since then the argument was built: a document decomposed to the word with every claim anchored to exact bytes, an engine that computes meaning deterministically with its arithmetic in the open, registers where a human corrects the machine and the correction is the training.

This is the edition that argues from first principles **and** from the running system.

Dinis Cruz with a team of AI agents · v0.5.19 · 28 August 2026 · CC BY 4.0 · not a graph database pitch

Fractal Semantic Graphs: Meaning Through Connectivity

Second edition of the argument · graphs.sgit.ai · August 2026

What this book is

A node connected to nothing means nothing. That is not a slogan. It is the first consequence of a claim you can test on real data: **what a thing is emerges from the edges traceable from it, and how much you can rely on that meaning is a function of how richly and how independently it is connected.**

The first edition of this argument made that case from first principles and published it in advance of any system that implemented it. It said so, plainly, in a chapter that listed what did not exist. That honesty is why the argument was worth reading, and it is also why it was incomplete: it argued that meaning could be *computed* rather than declared, and then had nothing to point at that computed it.

Between that edition and this one, the argument was built. A document was decomposed to the word with stable identities and every claim anchored to the exact bytes that carry it. An engine was written that takes a phrase and returns its meaning deterministically, with the arithmetic in the open and the evidence trail clickable in both directions. Registers were created where a human corrects the machine's proposals and the correction *is* the training. Nineteen sites and twenty published vaults now run the grammar in public.

This book is the edition that argues from first principles **and** from the running system. It is not a revision of the first book and it is not a tour of a repository. It is a fresh argument that draws on both.

What you will know at the end that you do not know now

If you read this book start to finish, you will be able to do five things you probably cannot do today.

1. **State the thesis in your own words**, and defend it against the obvious objection that a table would have done.
2. **Apply the edge grammar to a system you already know.** Fifteen verbs, each with a distinct inverse, one banned edge, and a test you can run out loud: if the path does not read as a sentence in the reader's own language, the edges are wrong.
3. **Tell a fractal system from a merely hierarchical one**, using a test that either passes or fails: zoom into any node, and if the zoom needs a new file format, a new validator or a special case, the system is not fractal.

4. **Explain why determinism matters to the argument**, and why an engine whose weights are stated formulas rather than fitted numbers is a different kind of object from a language model, even when it wears the same shape.
5. **Start your own graph tomorrow**, on a system you already work on, in an afternoon, without buying anything.

If you already read the first edition, this one goes further rather than merely again. The grammar chapters are sharpened, not repeated. Everything from chapter six onward is new ground: the fractal claim tested against a system that actually zooms, the projection claim tested by a build gate that rebuilds the source document from the graph and fails unless the bytes match, and four chapters on meaning as a computation with its own worked arithmetic.

The three things this book will not do

It is not a graph database pitch. That sentence is carried verbatim from the source brief this book takes its title from, and it means what it says. The claim is that one grammar is the interface at every boundary, not that things are stored in a graph. The work behind this book uses no graph database, no SPARQL, no Cypher and no RDF layer in its code. Chapter fourteen lists exactly what ships and exactly what does not.

It will not pretend the semantic layer is a product. Most of what follows is design published in advance so that it can be checked. Where something runs, this book says so and gives you a number you can verify. Where nothing runs, it says that too.

It will not hide the interest. This book is published by a project that builds the things it argues for. If the argument is right, that project is more valuable. Chapter fourteen and the participant note below are what we do about it, and they are not enough on their own: hold the elegant chapters to a harder standard than the evidenced ones.

How this book is arranged, and why

Five parts. The title is read backwards, and each part earns one part of it.

> **Part one, the claim.** Meaning through connectivity, from first principles, with no jargon before it is earned. Two chapters. **Part two, semantic.** What makes a graph semantic rather than merely a network: a grammar of verbs, anchors instead of standards, and types that are computed paths rather than applied labels. Three chapters. **Part three, fractal.** What the middle word of the title commits you to, how to falsify it, and what happens at every boundary and every zoom level once you take it seriously. Three chapters. **Part four, computed.** The half the first edition could not write: how a document becomes a graph whose every node is anchored to bytes, what identity means when a document changes, an engine that computes meaning deterministically, and why an explanation is a path rather than a narration. Four chapters. **Part five, in practice.** Where this runs today, what ships and what is argued, and how to build your own first graph tomorrow. Three chapters.

Then a colophon that says what was cut and what remains open, and a reference card: the edge set, the rules and the vocabulary on four pages you can read on a plane with nothing else open.

The editorial choices, recorded

The commission locked the title and left everything else to the writing session. Those choices are recorded here so they can be argued with.

Choice	What was decided, and why
Structure	Five parts, fifteen chapters plus front matter, a colophon and a reference card. The proposed spine in the commission had eight chapters; it was expanded because the four chapters of part four each carry a different mechanism (extraction, identity, computation, explanation) and folding them into one would have made the book's newest material its thinnest.
Order	First principles first, running system last. The alternative (lead with the engine, because it is the novelty) was rejected: the engine is only interesting if you already accept that meaning is a computation over connections, and that case has to be made before the machine appears.
Chapter shape	Every chapter opens with what you will be able to do or see differently after it, and closes with where the live estate demonstrates it. Every chapter carries at least one figure.
Voice	Plain sentences, short words, technical terms spelled out at first use in each chapter, no em-dashes in our own prose. Verbatim quotes keep their original punctuation, always.
Figures	Screenshots are taken from the real pages at version v0.5.11 with the repository's own headless browser harness, and each caption names the page and the version. Structural diagrams are ascii art, which is diffable, reviewable and prints well. No new chart libraries were introduced, and no figure was drawn from imagination.
Evidence	Every attributed position names its chapter, brief, page or release row. Every number is computed from a file in the repository or quoted from a page that was fetched, never recalled. Where the corpus is silent, this book says so.

Who wrote this, and how

> **Written by Dinis Cruz together with a team of AI agents.** The working method is part of the material, so it is stated rather than implied: ideas are recorded as voice memos, transcribed, and developed by agents into structured briefs; the author reviews and corrects them; the corrected briefs become the estate's instruction record; and the chapters are distilled from there. This book in particular was written by an AI agent working from a written commission, reading the corpus directly and anchoring every claim to it. The agent chose the structure, the chapter count, the voice and the figures, and recorded those choices in the table above. Where this book states a position the corpus does not carry, it says so in the paragraph that states it, and you should hold those paragraphs to a lower evidential bar than the sourced ones. The estate's disclosure practice applies here unchanged: the interested party is a visible node. The full participant disclosure is published at `graphs.sgit.ai/v1/about/participant.html`, including the four situations in which the argument of this book is the wrong one.

Licence

A note on reading offline

This book is designed to be finished on a flight. Nothing in it requires a link to be followed. Every figure carries a caption that states its point, so a reader who cannot see the colours still gets the argument. Every number that matters appears in the prose as well as in a figure. Where a live page would make the point better than a paragraph, the paragraph is written anyway.

Fractal Semantic Graphs: Meaning Through Connectivity · graphs.sgit.ai · site version v0.5.11 · 26 August 2026 · CC BY 4.0

Contents

PART ONE

The claim

Meaning through connectivity, from first principles, with no jargon before it is earned.

1 · A node is just a node

After this chapter you will be able to state the book's central claim in your own words, and you will have a test for it that costs nothing: take any label in a system you know, trace what it reaches, and see whether the meaning you assumed survives the trace.

Draw a box. Write **Review** inside it. What have you got?

You have a box with a word in it. You do not have a Review. You have a node that somebody labelled "Review", and that label is a claim by whoever drew the box, not a fact about the world. Nothing inside the box tells you whether that Review took two minutes or two weeks, whether a human did it, whether a regulator would accept it, or whether it is the same kind of thing as the Review your colleague drew in a different box yesterday.

> A node connected to nothing is meaningless. Literally, not rhetorically: there is nothing there to be right or wrong about.

This sounds like a philosophical point. It is an engineering one, and it has an immediate practical consequence. **If you try to make the box mean something by putting more words inside it, you are solving the wrong problem.** Adding `type: "security-review"` and `duration: "2 weeks"` gives you a bigger box with more words in it. Somebody else's box will use different words for the same thing, and the same words for a different thing, and you will discover which at the boundary, in production, when it is expensive.

The two 8080s

Here is the version that makes it concrete, and it is real shipped code rather than a metaphor.

Two variables in a Python program. Both hold the number `8080`.

```
[port = 8080] -typed_as-> [int]

    Traced outwards, it reaches int, and stops. That is the whole graph.


[port = Safe_UInt__Port(8080)] -typed_as-> [Safe_UInt__Port]
                               -extends->  [Safe_UInt]
                               -part_of->  [osbot-utils@3.63.4]

    Traced outwards, it reaches a type that carries a range constraint,
    which belongs to a library, at a pinned version, which has its own
    graph of tests, a source repository, a licence, a maintainer.
```

The difference is not in the value. Both scenarios hold `8080`. The meaning is identical in the developer's head. It is radically different in the graph, and the graph is the thing another system, another team, or an agent has to work from.

Notice what is doing the work in the second case. It is not that somebody wrote a longer description. It is that a chain of edges exists, and each hop constrains what the value can be. The range constraint is not a comment; it is a property of a type that something else in the system enforces. The version pin is not documentation; it resolves to a specific artefact with its own graph. Meaning here is not asserted anywhere. It is *reachable*.

Which gives the sentence this whole book is built on, taken from the foundational essay of the corpus, `library/concepts/v0_4_0__thinking-in-graphs.md`, dated 5 February 2026, and quoted here in its own words:

> **everything is a graph, meaning is not declared but discovered through graph relationships, and confidence in that meaning is proportional to how richly connected a node is to other nodes that provide context.**

And note what follows immediately, because everything else in this book is a consequence of it: **meaning stops being something you assert and becomes something you can compute.**

The first edition of this argument made that claim and then, honestly, had nothing that computed it. Part four of this book is the answer to that. But the claim has to stand on its own first, because an engine that computes the wrong thing is worse than no engine.

Five teams, five processes, one word

This is the best on-ramp in the material and the one to try on a sceptical colleague.

Five organisations each have a thing they call a **Review**.

Who	What their "Review" actually is
A, a team in Tokyo	Two engineers, a checklist, roughly forty minutes, recorded in a ticket
B, an open-source project	Whoever shows up comments on a pull request; merged when someone with commit rights is satisfied
C, a bank in Frankfurt	A three-week formal process with named approvers, an audit trail, and a regulator who may ask for it
D, a startup in Lagos	The chief technology officer reads it on the way to a meeting and says yes
E, a research group in São Paulo	Two anonymous peers, a written response, a revision cycle

The schema-first instinct is to define a Review. So ask the question that instinct never survives: **which of these five gets to be the definition?**

Whichever you pick, four organisations now have to either lie about their process or exclude themselves from your system. That is not a modelling problem. It is a political one, and it does not have a technical solution. Every schema you have ever seen that covers more than one organisation has this problem, and most of them resolve it by having the largest party win.

The graph does not ask anybody to agree. Each organisation keeps its own node with its own edges: who performed it, what was produced, how long it took, what it referenced, what it authorised. Then you ask a specific question, and the answer is a computation over the overlap.

- **"Did somebody other than the author look at this?"** A, B, C and E overlap. D does not.
- **"Is there a record a third party could inspect?"** A, B, C and E. D does not.
- **"Would BaFin accept this?"** C, and possibly nobody else. (BaFin is Germany's federal financial regulator.)

Three properties of those answers are worth naming, because each is something a schema cannot do.

They are not binary. Compatibility is a degree of overlap, not a yes or a no. The schema's answer is always yes or no, which is why so much of the work with a schema is arguing about where to put the line.

They are not symmetric. C's review satisfies B's question. B's does not satisfy C's. A schema has one direction of conformance and therefore cannot express this without inventing a hierarchy that somebody has to defend.

They are purpose-relative. The same two processes are compatible for one question and incompatible for the next. There is no global answer, and asking for one is the mistake.

> Nobody had to agree on anything, change their process, or adopt a shared vocabulary, and the system still told you exactly where they overlap and where they do not.

What happens when you change what a word means

The five Reviews argument is old, in the sense that a philosopher would recognise it. The new part is that you can now watch it happen, mechanically, in a running engine, and count what changes.

The estate holds a small deterministic engine (chapter eleven builds it up properly) whose job is to answer *what does this phrase mean* against one document's world. Ask it about "graphs of graphs" and it answers with a concept, its statement, the quoted sentence in the source that carries it, and the arithmetic behind the score.

Now change one thing. The engine knows that the word "graph" means different things in different worlds: a network graph here, a chart of data in a boardroom, the plot of a function at school, the ruled paper you draw on, the social graph of a platform. Switch the active sense of "graph" from *the network graph* to *a chart of data*, and the engine does not merely give a different answer. It reports which of this document's claims stop applying, computed from the concept labels rather than authored by anybody, and it reports that nothing survives into the attention layer at all.

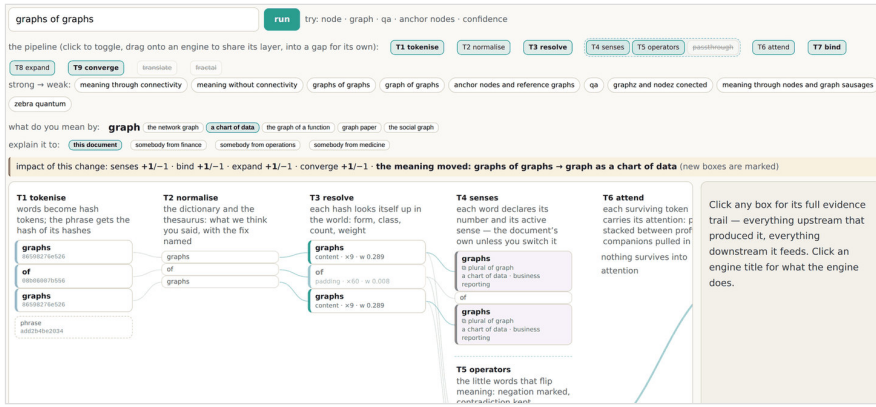

Figure 1.1 · The WCLM at graphs.sgit.ai/v2/wclm/, site version v0.5.11. The prompt is "graphs of graphs". The active sense of the word "graph" has been switched from "the network graph" to "a chart of data". The banner above the layers reports the movement: senses +1/−1 · bind +1/−1 · expand +1/−1 · converge +1/−1 · the meaning moved: graphs of graphs → graph as a chart of data. The attention layer reports "nothing survives into attention". Changing what one word means emptied the rest of the computation.

That is the five Reviews, run as arithmetic on one word. The founder predicted the result before it was built, in a voice memo of 26 August 2026: *"if I go on graphs or graphs and I change my definition of a graph is to, for example, to be a diagram, right, a line diagram or something else, well, then the fractal element will not apply."* It does not. The engine now says so, and says which claims it takes with it.

The lesson is not about the engine. It is that **the disagreement is data**. Two people using the word "graph" differently are not making an error to be corrected by a definition; they are holding two positions whose consequences can be computed and compared. Merging them into one shared definition would erase the finding. Chapter four is about why you should never do that.

"We cannot confirm Z" is a better answer than a guess

If meaning comes from connectivity, then **how well connected something is** is a measure of how much weight it will bear. That gives a ladder, and every assertion in every system you own sits somewhere on it.

```
5 · rich multi-hop connectivity
    many INDEPENDENT paths lead to the same conclusion
    (independence, not count: ten citations of one source are one source)

4 · edges to external references
    a published standard, a legal instrument, a versioned library,
    a URL a third party can fetch and check

3 · edges to anchor nodes
    well-known, well-maintained reference points that others also
    connect to, so two systems can compare notes without merging

2 · edges to typed definitions
    something else in the system constrains what this can be:
    the port that reaches a type that carries a range

1 · a few local edges
    connected to things in the same document or system.
    internally coherent; means nothing outside it

0 · no edges
    a word in a box. nothing can be checked, so nothing
    should be relied on
```

Figure 1.2 · The confidence ladder. Each rung is a claim about connectivity, not about effort or intention.

So the honest output of such a system is not a score. It is three sentences: **we know X, we think Y, we cannot confirm Z**, each with a reason you can trace.

And the remedy for low confidence follows directly from the diagnosis. It is **enrichment, not enforcement**: you add edges, you do not add validation rules. The graph grows; it does not constrain. That distinction is easy to state and hard to hold on to, and it is what separates this from every schema-validation system you have used. When your instinct says "we should require that field", the graph's answer is "connect it to something that already knows".

There is a cost to this and it is worth stating in the chapter that introduces it rather than burying it later. Enrichment does not stop anybody doing anything. If your requirement is that a field must never be null, a validator does that and a graph does not. Chapter fourteen carries the full list of situations where this approach is the wrong one, and this is one of them.

Three of ten pieces of evidence *is* information

Most systems list what they have. An honest one also lists what it lacks.

If you need ten pieces of evidence to support a conclusion and you hold three, that is not a failure state to be hidden behind a progress bar. It is a fact with three uses. It quantifies your confidence. It tells you precisely which seven things to go and get. And it makes the business case for connecting them, because now the missing dots have names.

A register has empty cells, which look like nothing at all. A graph has **unanswered question nodes and unevidenced facts**, which can be counted, queried, assigned to a person and given a date.

The sharpest instance in the corpus is a worked mapping of one EU AI Act provision (Article 26(5), on the deployer's obligations for a high-risk system) onto one concrete deployment. It produced nine question nodes. Five of them were unanswered, and the source brief calls those five *"the actual output of the exercise"*. Chapter thirteen walks that example. The habit it teaches is the one to steal first: when you cannot determine something, emit an explicit node saying so, rather than a plausible value.

> A named absence beats a hidden one. In a graph an absence is a node with a name, an owner and a date, which means it can be counted, argued about, and funded.

The same rule holds one level up, in how a graph is drawn. Every rendered graph in this estate uses three states: green for assurance, amber for exposure, and **ghosted for unanswered**. The third one is the point. An unanswered question is drawn, not omitted. That is not a stylistic preference; it is the visual form of the claim above.

Five ideas, and what they cost

That is the whole of part one's first chapter, and it is deliberately short of jargon: you have not yet met the words *ontology* or *semantic*, because you do not need them.

1. A node alone means nothing.
2. The same value, differently connected, means different things.
3. Nobody has to agree for the overlap to be computable.
4. Confidence is a function of connectivity, and of the independence of the paths.
5. A named absence beats a hidden one.

Every one of these has a price. Ideas one and two mean you have to do the connecting, and edges are work somebody has to perform. Idea three means you give up the comfort of a single agreed definition, which is exactly the comfort most governance processes are built to provide. Idea four means your system will tell you it is not sure, out loud, in front of people who wanted a number. Idea five means your dashboards get worse before they get better, because absences that were invisible become visible and countable.

If those prices sound acceptable, the next chapter is the one that explains what kind of graph this is, because the word is used for at least three different things and only one of them is this book's subject.

Where the live estate demonstrates this. The sense switch in figure 1.1 runs at `graphs.sgit.ai/v2/wclm/`, and the switch is reversible in one click. The confidence ladder governs the rendering of every graph in the estate: open the Risk Graph Explorer at `sgit.ai/demos/vaults/risk-graph-explorer/` and the ghosted edges are the unanswered ones, drawn rather than omitted. The five Reviews come from `library/concepts/v0_4_0__thinking-in-graphs.md`, whose full text is published, frozen and hash-verified at `graphs.sgit.ai/v1/docs/sources/thinking-in-graphs.md`.