Episode Book

EPISODE #5949

Inside Deep Research Agents: Claims, Citations, and the 84.7% Orchestrator Problem

PUBLISHED	14 October 2026 at 08:41 UTC
DURATION	23:36
LISTEN	myweirdprompts.com/episode/deep-research-agent-claim-ledger-93879/
HOSTS	Corn, Herman, Hilbert
SCRIPT MODEL	deepseek-flash
VOICES	chatterbox-regular (chatterbox-local)
PIPELINE	V5.4
BOOK GENERATED	19 October 2026 at 17:41 UTC

SYNOPSIS

What actually separates a research agent that grounds its claims from one that piles up web searches and writes a confident summary? This episode breaks down the four-stage deep research pipeline — planning, retrieval, verification, synthesis — and the three failure classes that survive every surface check: unsupported claims, silent coverage gaps, and unbounded drift. The fix isn't more search tools. It's treating research as a claim ledger under budget: a question graph with explicit statuses, an evidence ledger that stores verbatim spans instead of summaries, and a claim table where every atomic claim maps to supporting and contradicting evidence. We dig into why 84.7% of citation errors in agentic deep research systems originate at the orchestrator — not the searchers — and what LangChain's layered stack (LangChain, LangGraph, Deep Agents) means for where you place verification structure.

TRANSCRIPT



Corn

Daniel wants to know what actually separates a research agent that grounds its claims from one that does a pile of web searches and then writes a very confident summary. And underneath that, two questions. The architecture one, which is LangGraph, which is Deep Agents. And the trust one, which is retrieval, verification, citation, and what you do when two sources disagree.



Herman

The thing nobody building these trusts is the output.



Corn

Right. He also wants the split. How much of the grounding do you engineer into the workflow yourself, versus what you hand to the model or get free from the framework. Which is the harder question of the two, honestly.



Herman

Start with what the system actually is, because the word "agent" is doing a lot of hiding in there.



Corn

Define it, then. Because "agent" has become one of those words that means whatever the person selling it needs it to mean.



Herman

A deep research agent takes a question, plans a set of sub-questions, retrieves from the web, verifies what it found, and synthesizes a long cited report. Four stages. Planning, retrieval, verification, synthesis. That pipeline shows up in the 2025 survey of autonomous research agents and it's basically the same shape in every product on the market, which tells you it's the right decomposition.



Corn

And the failure that matters is what happens between stage two and stage four.



Herman

That's the whole episode. There's a weak mental model and a strong one. The weak model is: search more, then write. More queries, more pages, more context, and the writing is where you spend your effort. The strong model is: research is a claim ledger under budget. You're not accumulating text. You're accumulating claims, each one bound to an evidence span that supports it, and you're spending a fixed budget to close the gaps.



Corn

A ledger. That's an accounting frame, not a search frame.



Herman

Deliberately. Because the failure isn't that the agent didn't find enough. It's that the report contains things the evidence doesn't actually support.



Corn

Walk me through what that looks like, because "the citation is wrong" covers four different problems.



Herman

Three failure classes, and they're different. First, unsupported claims. A citation exists, it points at a real page, and the span doesn't entail the claim. That one survives every surface check you can run. The link resolves. The source is real. The sentence is plausible. Nobody catches it except a reader who goes to the page and reads the paragraph.



Corn

Only believed. Like a paragraph-level citation. Nobody can verify one of those. You can only take it on faith.



Herman

Which is exactly the line — a paragraph-level citation can't be verified, only believed. Scale that to a report and the citations are decorative.



Corn

What's the second one.



Herman

Coverage gaps. Silent omission. The agent didn't find the contradicting paper, and it also didn't tell you it didn't find it. There's no hole in the output. There's a hole in the world.



Corn

And the third.



Herman

Unbounded drift. Searching feels like progress. You've got a loop that keeps issuing queries, the loop is productive-looking, it's burning tokens, and the report is no better than it was twenty calls ago.



Corn

I've watched that from the inside. You keep going because stopping feels like giving up, not because you found anything.



Herman

And the point about all three is that they're architectural. They're not a model-quality problem that a better checkpoint fixes.



Corn

Hm. Say more about that.



Herman

When synthesis precedes verification, the generator optimizes for fluency. That's the trap. The writing model produces a smooth paragraph, the citation gets attached after the fact because the format requires one, and the citation is decoration. The summarization step is where provenance dies, and it dies quietly.



Corn

Because summarizing is the natural thing to do. You read six pages, you keep the summary, you drop the pages.



Herman

And now the model is citing from memory. Memory is not a citation.



Corn

So what's the actual mechanism. What does the ledger contain.



Herman

Three structures, and they're all just state that lives outside the model. First, a question graph. Your sub-questions, each one carrying an explicit status. Open, answered, blocked, abandoned. The status is the point. It's what stops drift, because the orchestrator can see whether it's actually closing anything.



Corn

Second.



Herman

An evidence ledger. One record per retrieved span. Store spans, not summaries. The span is the verbatim stretch of the source page that matters, kept in full.



Corn

Third.



Herman

A claim table. Atomic claims, each mapped to the IDs of evidence that supports it and evidence that contradicts it, plus a verification status.



Corn

Atomic is the load-bearing word there, isn't it.



Herman

It's the hinge the whole thing turns on. The unit of control is the atomic claim. One assertion, one testable proposition. If your unit is the paragraph, verification is impossible, because a paragraph contains five claims and three hedges and a transition, and there's no yes-or-no question you can ask about it.



Corn

So you shatter the report into single sentences that are either supported or not.



Herman

And then verification becomes mechanical. Three separate checks, not one. Entailment: does this span actually support this claim. Answer yes or no, and quote the sentence. Corroboration: do independent sources agree. Freshness: is the source current enough for the claim being made.



Corn

Corroboration is where people get smug. Two sources, therefore confirmed.



Herman

Two pages quoting the same press release are one source wearing two URLs. It's one origin with two addresses. If both trace back to the same wire copy, you have corroboration in the count and none in the fact.



Corn

That's a much nastier problem than fabrication.



Herman

It is, because fabricated citations mostly got solved. Modern retrieval systems largely kill the invention of papers that don't exist. What survives is misattribution. The paper is real. It's famous. It just doesn't say what the report claims it says.



Corn

The citation is genuine and the claim is wrong.



Herman

Which is why accurate citation needs a shift from similarity search to claim-level reasoning. Retrieving something that looks relevant is a different operation from checking whether it entails your specific assertion.



Corn

Give me the mechanical failures. The boring ones that actually happen.



Herman

Three logged ones. Format compliance without semantic binding is the big one — the marker is present, the binding is absent. Index drift: in a multi-turn run, you re-index the source list and now "[2]" is ambiguous, because "[2]" meant one page in turn three and a different page in turn seven.



Corn

So old citations silently repoint.



Herman

They do. Which is why you don't cite by position. You cite by stable ID. And the third is summarized sources — the agent cites a secondary summary instead of the primary document, because the summary was what was in its context.



Corn

So the fix.



Herman

Force citations to be structured objects. Not a marker and a URL. An object with three fields: the claim, the source ID, the span. Then render the report by joining those objects to the prose. Any claim without a mapped source ID gets dropped, or rendered with a visible flag.



Corn

And then a separate pass.



Herman

A separate verification pass whose only job is to answer yes or no with a quote. It doesn't write. It doesn't summarize. It reads the claim, reads the span, and votes.



Corn

Here's the part that reframes the lot. The errors don't come from where people think.



Herman

They don't. There's a study that localizes every citation error in an agentic deep research system to the agent that introduced it. Eighty four point seven percent of the errors in the final report originate at the orchestrator.



Corn

Not the searchers.



Herman

The orchestrator. Roughly a third of those are outright hallucinations, and the rest are citation mistakes. The synthesis and handoff step is the weak link.



Corn

That's the opposite of the instinct. Everyone's instinct is to add more search tools, add a second provider, give the researcher more firepower.



Herman

And the empirical answer is that your researcher is fine. Your researcher did its job. Something got mangled on the way from twelve subagent reports into one document, and that's where 84.7% of the damage is.



Corn

"Something got mangled" undersells how specific that is. The orchestrator is the weakest link, and yet the verification pass usually runs at the orchestrator's own layer, where its context is already saturated.



Herman

Which cuts against the instinct to add more search tools.



Corn

It cuts against the entire premise of building a bigger retrieval arm.



Herman

And two fairly simple interventions lifted citation recall by five percent without hurting quality. Small changes, applied in the right place. Not a bigger model.



Corn

So the ledger isn't about retrieval at all. It's about protecting the handoff.



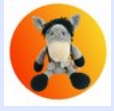
Herman

That's the reframe. Every claim that travels is a claim that can get corrupted, so you make it travel as a structured object with its evidence attached, and you verify at the seam.



Corn

So if the weak model is search-then-write and the strong model is a claim ledger, the question is what happens when you try to force verification structure into a workflow that's being built on layers of abstraction — and specifically whether that structure helps or just clogs the pipeline.



Herman

That's the framework question. LangChain's stack is layered now, and the layers matter for where you'd put any of this.



Corn

Define the layers.



Herman

LangChain is the building blocks. Models, tools, prompts, the standard primitives. LangGraph is the runtime. Durable execution, streaming, human-in-the-loop interrupts. If your agent needs to survive a process restart or pause for a human, that's LangGraph's job.



Corn

And Deep Agents.



Herman

Deep Agents is a harness. It's the same tool-calling loop you'd write yourself, plus a set of built-in capabilities. Four categories. Execution environment — tools, MCP, a virtual filesystem, permissions, sandboxed code execution. Context management — skills, memory, summarization and offloading, prompt caching. Delegation — subagents, plus task planning. And steering — human-in-the-loop interrupts.



Corn

Delegation is the one that maps onto the ledger.



Herman

It does, and there's a detail that matters. Subagents run in isolated context windows by default. Mode "isolated." They get their own window, they don't inherit the parent's history. There's a "fork" mode for when you do want them to inherit the parent's full history.



Corn

And they're stateless.



Herman

Stateless, and they return a single final report. Heavy subtask work stays isolated and comes back compressed. That's the architecture that keeps your orchestrator's context from filling up with six pages of scraped text.



Corn

Which is what makes the orchestrator the weak link in the first place. It's the one agent that has to hold everything.



Herman

Right. And the compression on the way back is where claims get lost or mutated. The subagent had the span. The orchestrator gets the summary.



Corn

There it is.



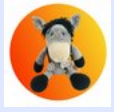
Herman

Task planning is opt-in as of version zero point seven. You have to pass the todo list middleware to get the write-todos tool. Earlier versions had it on by default.



Corn

That's a small detail that tells you where the framework's philosophy is going.



Herman

It does. The default is getting leaner. They're making you ask for structure.



Corn

And the official tutorial?



Herman

The LangChain deep research tutorial builds almost exactly the ledger shape. A deep agent orchestrator with a research-agent subagent. A search tool that fetches the full webpage as markdown, not just snippets. And a workflow prompt with explicit steps: plan, save the request, research through subagents, synthesize, write the report, verify.



Corn

Full markdown instead of snippets. That one's doing real work.



Herman

It is. A snippet is a summary made by somebody else's ranking system. You can't verify a claim against a snippet because you don't have the surrounding paragraph, and the surrounding paragraph is usually where the qualification lives. "The effect was significant in mice" is in the sentence above.



Corn

And now the counter-argument, because there's a credible one.



Herman

There is. Lance Martin wrote up his own evolution on open deep research, and it's essentially a case for removing structure. He started with an orchestrator-worker workflow in late twenty twenty four that deliberately avoided tool calling, because tool calling wasn't reliable enough. Then he moved to a multi-agent system. Then he moved the writing to a single final step.



Corn

And the score.



Herman

Forty three point five on Deep Research Bench. Top ten.



Corn

After he removed things.



Herman

His three lessons: understand your application structure, re-evaluate structure as models improve, and make it easy to remove structure. He cites Hyung Won Chung — add the structures needed for the level of compute and data available, remove them later, because the shortcuts will bottleneck further improvement.



Corn

So the short version is: every scaffold you build is a bet that the model can't do this yet, and the bet expires.



Herman

That's the bitter lesson. And he still uses LangGraph. He just sticks to the low-level building blocks, nodes and edges, the parts he can reconfigure without a rewrite.



Corn

So here's the tension I want to sit on. Those two arguments are pointed at each other.



Herman

They are.



Corn

One says: engineer explicit verification structure, because the model can't be trusted to ground itself. The other says: remove hand-imposed structure, because it bottlenecks the next model.



Herman

Both recent. Both credible. Both from people who shipped something.



Corn

So do they actually conflict?



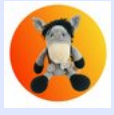
Herman

My guess is not entirely, and the resolution is that verification structure is different in kind from workflow structure. Those are two different things that look like one thing.



Corn

Unpack that.



Herman

Workflow structure is a bet about capability. You route a task through three fixed steps because you don't trust the model to figure out the steps. Better model, the routing is dead weight, and it actively hurts, because it constrains a planner that's now better than your flowchart.



Corn

And verification structure is a bet about what.



Herman

It's a bet about the goal. It's not a scaffold that helps the model do something it can't do yet. It's a definition of what counts as done. You're not saying the model is too stupid to check itself. You're saying the output has a property — every claim traceable to a span — and that property is the product, not the process.



Corn

So it doesn't expire, because it's not a capability claim.



Herman

That's the argument. But nobody's cleanly separated the two yet, which is why it's a real tension and not a resolved one. The honest version is that the ledger might be workflow structure wearing a verification costume, and we won't know for a couple of model generations.



Corn

Which is an uncomfortable place to leave it, and I think it's the right one.



Herman

One thing that does cut through it: the cost model.



Corn

Because cost is where it gets concrete.



Herman

Verification scales with claims, not searches. A run that searches fifty times and produces twelve claims pays for twelve verifications. Fifty searches, twelve verifications. The expensive thing is the writing and the checking, not the querying.



Corn

That flips how you budget a run.



Herman

Completely. If you think cost scales with searches, you cut searches, and you get coverage gaps, which is failure class two. If you know cost scales with claims, you manage the claim count and the verification depth, and you constrain iterations and wall clock instead. Exalted AI's suggested caps are forty search calls, sixty verification calls, fifteen minutes of wall clock, and six iterations.



Corn

Fifteen minutes on a wall clock. That's the one people will hit first.



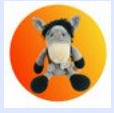
Herman

Iterations, I'd say. Six is generous and drift is patient.



Corn

And the benchmarks.



Herman

DeepResearch Bench. A hundred PhD-level tasks across twenty two fields. Two scores: RACE for report quality, FACT for citation reliability.



Corn

Give me the citation accuracy spread, because that's the interesting column.



Herman

Perplexity holds the highest citation accuracy at ninety point two four. Gemini at eighty one point four four. OpenAI at seventy seven point nine six.



Corn

So there's a twenty point spread between the top product and the third one, on the thing this whole episode is about.



Herman

And the caution is the evaluator. DeepResearch Bench is migrating from Gemini two point five Pro to GPT five point five as the judge, which makes scores incomparable across versions. The current leaderboard top RACE is fifty eight point zero three, and if you're comparing that to a number from a year ago, you're comparing two different measurements.



Corn

The yardstick moved.



Herman

There's also a result worth flagging because it cuts against everything I've said: AI21 got to number one on DeepResearch Bench two by merging the outputs of weaker agents. Total score sixty four point three eight. Not by having the best single agent — by orchestrating several mediocre ones into one better answer.



Corn

Orchestration beats quality.



Herman

At that layer. The aggregation step is doing real work.



Corn

And then the sobering one.



Herman

The sobering one is retrieval. TaxoBench measures whether agents actually retrieve the papers that experts cite. The best agent retrieves twenty point nine two percent.



Corn

Twenty point nine. Out of a hundred.



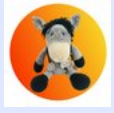
Herman

The best one. Which means the ceiling on how grounded any of this can be isn't the synthesis layer at all — it's that the system never found the right document in the first place. You can have a perfect claim ledger and perfect verification, and you can only verify against the evidence you retrieved.



Corn

You can't verify what you never read.



Herman

You can flag it as a coverage gap. That's the honest move. But the report is still built on twenty out of a hundred.



Corn

To Daniel's point, though — the ledger is what lets you know it's twenty out of a hundred instead of believing it's a hundred.



Herman

That's the whole argument in one line, and it's the reason the ledger survives even a terrible retrieval rate.



Corn

You build the ledger and you catch the gaps.



Corn

This whole thing is about where the claim lives. What carries the evidence.



Hilbert

I was on the span.



Hilbert

Three forty. That's the number. We capped it at three hundred and forty spans per report, and all of it fell over at three forty one. Not gracefully. It didn't degrade. It stopped.



Herman

What was the span in your setup?



Hilbert

The actual sentence, plus the four sentences around it, stored with its byte offset into the original document. The offset was the only thing that made the second pass possible. You re-fetch the document, you seek to the byte, you confirm the sentence is still there. If it isn't, the span is stale and the claim gets flagged. That was the whole re-verification loop. Two hundred lines of code and a hash.



Corn

Byte offsets instead of paragraph numbers.



Hilbert

Paragraph numbers shift. Layout changes, the ad block moves, suddenly paragraph fourteen is paragraph fifteen. A byte offset into the raw response doesn't move unless the document itself changed. Nobody believed me on that until the third rebuild.



Herman

How did the count get to three hundred and forty in the first place?



Hilbert

Because the same sentence was in twelve documents with twelve different surrounding contexts. Same sentence, twelve spans, twelve verifications. You can't dedupe them, because the context is what you're verifying against. The sentence that is a hedge in one paper is a finding in another because of the paragraph under it.



Corn

So the count explodes on the source side, not the claim side.



Hilbert

It does. And the cap was arbitrary. Three forty was where the nightly job stopped finishing before the morning. We never found out what happened above it.



Herman

The corpus must have been enormous.



Hilbert

The corpus was one aggregator. One. We found out in month four. Every span traced back to the same wire service that had summarized everything first, and we had been verifying against a summary the entire time.



Corn

Twelve URLs, one source.



Hilbert

Precisely the thing you two spent the first half of the show describing. We built the whole apparatus and pointed it at a mirror.



Herman

And the client group didn't know.



Hilbert

They had never read a primary source. Bright people, every one, and not one had ever opened the original document. They'd built a pipeline on top of summaries and were paying me to tell them the summaries were accurate.



Corn

What did you tell them.



Hilbert

The truth, and they didn't like the invoice, so they paid me in something that wasn't money. I'm still not sure what it was.



Herman

That's a real constraint though. Byte offsets at the span level would catch the mirror problem, because twelve URLs pointing at one aggregator would share a document hash.



Hilbert

That's what I said. They asked for a status update instead.



Corn

The thing Hilbert just described — twelve URLs, one source, every span tracing back to the same summary — that's the failure mode we opened with, which is a decent place to leave it.



Herman

The open question is whether verification structure is different in kind from workflow structure, or whether it's workflow structure wearing a better suit. If it's the second one, the bitter lesson applies and it comes out in two generations. If it's the first, it's permanent, because it's a definition of the output rather than a crutch for the model.



Corn

And nobody has separated the two cleanly yet.



Herman

Not cleanly, no. And the retrieval ceiling is the honest limit underneath all of it. Twenty point nine two percent. You can build a perfect ledger and you're still verifying against what the agent managed to find.



Corn

Which is why the ledger matters more, not less. It's the only thing that tells you how much you're missing.



Herman

Thanks as ever to our producer Hilbert Flumingtop, who has opinions about byte offsets.



Corn

If you want more of this, try episode twenty, Architectural AI; episode sixteen, On Deepfakes, SynthID, And AI Watermarking; and episode nine, Benchmarking Custom ASR Tools - Beyond The WER. This has been My Weird Prompts. If you want to send us a prompt of your own, it's t dot me slash MWP listener bot.



Herman

We'll be back soon.



Corn

See you tomorrow.