Episode Book

EPISODE #6043

Write the Spec Before You Pick the Model

PUBLISHED	16 October 2026 at 21:43 UTC
DURATION	33:48
LISTEN	myweirdprompts.com/episode/workload-spec-before-model-selection-45306/
HOSTS	Corn, Herman, Larry, Hilbert
SCRIPT MODEL	deepseek-flash
VOICES	chatterbox-regular (chatterbox-local)
PIPELINE	V5.5
SOURCES	14 cited by the research (listed at the end)
BOOK GENERATED	19 October 2026 at 17:42 UTC

SYNOPSIS

Picking a model starts long before you open a leaderboard. This episode works through how to write a workload spec: the input distribution, the output contract, the success criterion, and an asymmetric failure-cost matrix that says which model's mistakes are cheaper for you. Along the way: AWS's four failure modes of judging on vibes, Anthropic's specificity test, rule-based checks versus judge-scored rubrics, and why thirty eval items can't tell a 92 from a 96. If you've ever chosen a model because it sounded confident, this one's for you.

TRANSCRIPT



Corn

Herman, before we start, I want to say something about Daniel's question, because I think it's the kind of question that sounds boring until you've watched a team get it wrong.



Herman

It sounds like a documentation question.



Corn

It is a documentation question. That's why people skip it. Daniel's asking how you actually put together the document that says what your workload requires before you go shopping for a model. And he named our own workload as the example, which I appreciate, because ours is unusually annoying. Long context, rich multi-character support, and cost-effectiveness at scale.



Herman

Three requirements that pull against each other.



Corn

Constantly. And the point of his question is that you cannot run an eval, or pick a model, until you've written down what you're evaluating for. Which is the step everyone treats as the boring preamble.



Herman

The boring preamble is the whole job.



Corn

So where does the document start?



Herman

It starts before it starts, if that makes sense. There's a consistent position across most of the serious guidance on this now, and it boils down to one sentence. You cannot measure good until you have said what good means. That sounds like a fortune cookie until you watch what happens without it.



Corn

Describe what happens without it.



Herman

You get what AWS called judging on vibes. They wrote this up last year, and the four failure modes they name are worth just listing, because I've committed all four. Subjective bias, where the tester favors whichever answer sounds more confident or uses more exotic words, regardless of whether it's correct. Lack of coverage, where you test with a handful of prompts you thought of yourself and never touch the edge cases. Inconsistency, where two evaluators disagree because they care about different things. And no trackable benchmark, so when you change a prompt or a model you can't tell whether accuracy went down.



Corn

Exotic words. You are the target audience for that one.



Herman

I am aware.



Corn

You once told me a model was better because it wrote "nonetheless" more often.



Herman

I said it had a more formal register. There's a difference.



Corn

There isn't.



Herman

There's a small difference, and it doesn't matter to this point. The point is that all four of those breaking points are documentation failures. If you'd written down what a correct output looks like, bias gets caught by the criterion. If you'd written down the input distribution, you'd know you're only covering a fraction of it. The document is what makes the eval possible.



Corn

Okay, and Daniel's asking for the practical version. What actually goes in the document.



Herman

Right. So the framing I'd start from is that model selection is a procedure, not a vibe. You write a task definition with four parts. The inputs, which is what actually arrives, including the messy ones. The output contract, which is what shape the answer has to come back in, whether that's free text or a fixed set of labels or a structured object with named fields. The success criterion, which is what a correct output is and how you'd know. And the cost of failure, which is what happens when it's wrong.



Corn

That last one is where most documents stop being useful, I think.



Herman

It's where they stop being written. Everyone writes down the happy path. Nobody sits there and works out what a wrong answer costs, and that's the part that decides which model you buy.



Corn

So before we build ours, let's get the trap out of the way, because it's the same trap Daniel's question is pointing at. A high ranking on a public benchmark doesn't mean the model is right for you.



Herman

AWS were blunt about that. Public benchmarks narrow the field, but they measure generalized performance. They can't see your organization's constraints around latency, or cost, or safety requirements. A model can top a general leaderboard and be excellent at trivia and useless with your industry terminology.



Corn

Or in our case, excellent at trivia and bad at keeping two characters in a scene from swapping personalities.



Herman

Which no leaderboard tests for.



Corn

Nobody scores that.



Herman

Nobody scores it, and that's exactly why the document has to exist.



Corn

Put the one-page version on the table. If Daniel's listening and wants to actually do this, what does the page look like?



Herman

Four things on one page. The input distribution, the output contract, preferably written as a schema, the success criterion in a form a grader could apply without you in the room, and an asymmetric failure-cost matrix.



Corn

Unpack the asymmetry one, because that's the sharpest idea in the whole build.



Herman

It says the obvious thing that nobody acts on. A model that is more accurate overall can be worse for you if its errors land on the expensive side. So you don't ask which model is better. You ask which model's mistakes are cheaper for you.



Corn

Give me the concrete version.



Herman

Suppose one model has a slightly better average score, but when it's wrong, it's wrong in the one category you can't afford, and the other model is slightly worse on average but spreads its errors evenly. The one with the worse average is the one you want.



Corn

Every procurement conversation I've ever sat near has gone the other way, because averages are easy to put in a slide and error classes are not.



Herman

And that's the entire argument for writing this down before you buy anything.



Corn

So build the elimination step first. What kills a model before you've tested anything.



Herman

Constraints. Latency, cost, context, modality. And the trick is that each one has to become a number, because a requirement that isn't a number can't eliminate anything. Take latency. You can't say you need it to be fast. You translate it into two numbers at two percentiles. Time to first token, and total completion time, both at the fiftieth percentile and the ninety-fifth, at your real prompt length, not a short one.



Corn

Why both percentiles?



Herman

Because the median is what you feel most of the time and the ninety-fifth is what you remember. A model that's fine at the median and terrible at the tail is a model that ruins a handful of generations a week, and you'll never spot it in a demo of five prompts.



Corn

And volume?



Herman

Volume becomes tokens. Requests per day times the input plus output tokens per request. And the thing people leave out of that arithmetic is that your system prompt is paid for on every single call, and that retries and tool loops multiply it, because a retry is another full trip through the prompt.



Corn

That's the kind of sentence that sounds obvious and still shows up wrong in every estimate I've seen.



Herman

It's the difference between a cost model and a guess.



Corn

Give me the specificity test, because that's the part I'd hand to somebody as an exercise.



Herman

Anthropic's docs have the cleanest version. You take a vague requirement and you refuse to accept it until it's measurable. So instead of "good performance," you write "accurate sentiment classification." Instead of "safe outputs," you write "fewer than zero point one percent of outputs, out of ten thousand trials, flagged for toxicity by the content filter."



Corn

One in ten thousand. That's a number that tells you how many you have to run to find out.



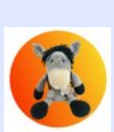
Herman

Which is the point. They have a worked example of the whole spec for a sentiment task. F1 of at least zero point eight five on a held-out set of ten thousand diverse posts, ninety-nine point five percent non-toxic outputs, and ninety-five percent of responses under two hundred milliseconds.



Corn

So that's a model requirement written as a sentence you could hand to a vendor and say yes or no.



Herman

You could put it in a contract.



Corn

The other framing I want on the table is the one that calls this a workload contract.



Herman

That's the aaai framing, and it's useful because it reads like something a lawyer would draft. A written statement of which tasks matter, what counts as an accepted result, which hard constraints apply, and the limits on quality, cost, latency, availability and operational risk. And it carries the right warning alongside it, which is that there is no universal winner across workloads.



Corn

Say that again for the people who want one answer.



Herman

There is no universal winner across workloads. The model that's best for transcription is not the model that's best for long-form conversation. Anyone selling you a single best model is selling you the thing that was best for their workload.



Corn

Which brings up sizing the eval set, because that's part of the document, not an afterthought. How many items?



Herman

A hundred to three hundred real items, stratified across the cases you care about. Not invented examples, actual ones from the input distribution. And the number matters, because the confidence interval is doing math you'd rather know about in advance.



Corn

Give me the two ends of it.



Herman

Two hundred items at ninety percent accuracy gives you about two points of uncertainty either way. Thirty items gives you five points. So if you run thirty items, you cannot tell a ninety-two from a ninety-six. You will make a confident decision on noise.



Corn

And people run thirty because it's a Tuesday and they want to be done.



Herman

They run thirty, they get a number, they ship. And then they wonder why the model that won the eval doesn't feel better in production.



Corn

So that's the constraints half. Now the split Daniel's actually asking about. Mapping the requirements onto measurable targets.



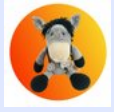
Herman

The clean version is that objective criteria get rule-based checks and subjective criteria get a judge. Objective means you can write code for it. Format validation, JSON structure, field presence, length thresholds, keyword presence, exact match against a known answer. Subjective means relevance, helpfulness, safety, tone, factual grounding, and for us, character voice.



Corn

Rule-based checks are the good half. They're cheap and they don't have opinions.



Herman

They're cheap, they're deterministic, and they fail loudly. If the structured output is supposed to have six named fields and the model returns five, you know instantly, and you don't need a judge to tell you.



Corn

And they never tell you whether the answer was any good.



Herman

Never. A response can pass every structural check and be completely wrong.



Corn

So how do you make the subjective ones measurable? Because "rich multi-character support" is not a number.



Herman

Two techniques. Likert scales, which is just a rating from one to five with the ends anchored. Rate coherence from one, nonsensical, to five, perfectly logical. Once you've anchored the ends, two people reading it mostly agree, and more importantly, you can average it across a hundred items.



Corn

And the other.



Herman

Expert rubrics. You write down what a four looks like and what a two looks like for each dimension, in words, so a grader can apply it without asking you what you meant.



Corn

Give me the AWS version of the move, because it's a good before-and-after.



Herman

Before, your eval reads "I like this answer more." After, it reads "Model A scored four out of five on correctness and five out of five on completeness." Same judgment, expressed so it can be audited.



Corn

And their six dimensions are a decent ready-made skeleton. Correctness, completeness, relevance, coherence, following instructions, format.



Herman

It's a good default because it's boring. Correctness can use exact match or a similarity score when you know the right answer. But for open-ended responses you need a human or a model doing the judging, because there is no string to compare against.



Corn

Which is where I'd want to stop and note the thing Daniel's whole question is really about. The document has to say which of these you're using and why, because the choice changes the answer.



Herman

It changes the answer more than most people expect. And that's the part I want to spend the rest of this on, because it explains the failure we've had.



Corn

Your setup.



Herman

My setup. I deployed four models against the same prompt set last month, scored them, and three of them were within a point of each other, which told me nothing except that I'd built a bad eval.



Corn

That's a productive kind of failure, though.



Herman

It's the only kind I reliably produce.



Corn

What was wrong with it?



Herman

I'd written the prompt set, but I hadn't written down what a correct answer looked like, so my scoring was me reading output and feeling good or bad about it. That's the vibes problem. I was the biased tester with no coverage and no consistency.



Corn

And you knew the four pitfalls.



Herman

I knew them in the abstract and violated all of them in one afternoon.



Corn

Let's get concrete about the three requirements, then. Start with the one that looks objective and isn't. Long context.



Herman

Long context looks like the easiest requirement in the world, because there's a number on the spec sheet.



Corn

And the number is not the number you get.



Herman

It never is. Every model advertises a window now, a hundred and twenty-eight thousand tokens, some of them a million. And the research on how models actually use that window is fairly brutal.



Corn

The middle.



Herman

The Lost in the Middle finding is that the performance curve is U-shaped. Models retrieve well from the beginning of the context and well from the end, and degrade in the middle. So a fact buried on page eighty of a long document is the hardest thing in the world for the model to find, and that's the position most of your context is in.



Corn

And there's a paper arguing it's worse than a retrieval problem.



Herman

There is. The argument is that performance degrades with length even when retrieval is perfect. So it isn't only that the model failed to find the right passage. Even when you've confirmed it found the passage, the length itself costs you accuracy.



Corn

That's a different failure. That's not a search problem, that's a comprehension problem.



Herman

And the benchmark critique comes from the same place. The argument against the common long-context benchmarks is that they don't separate the model's long-context ability from its baseline knowledge, so a model can score well because it already knew the answer, not because it handled the length. And they use fixed input lengths, so they can't tell you when a model breaks down.



Corn

When. That's the number we need.



Herman

When is the whole requirement. If I could only put one sentence in our requirements document about context, it would be this. We don't need the largest window. We need to know at what token count our quality falls off.



Corn

And there's a number that makes that visceral.



Herman

Web agent benchmark, failures in long context. Success rates drop from forty to fifty percent at baseline down to under ten percent at twenty-five thousand to a hundred and fifty thousand tokens. Across four different model families. And the snag is not the model refusing. It's the agent getting stuck in loops and losing track of the original task.



Corn

Forty down to under ten. That's a different model at a different length, it's the same model.



Herman

Same model, longer input.



Corn

So our eval target for context isn't "can it accept two hundred thousand tokens." It's a curve. We run the same test at ten thousand, at fifty thousand, at a hundred and fifty, and we plot where it breaks.



Herman

And that's a completely different document than the one most teams write. Most teams write "context window: one hundred twenty-eight thousand." Ours says "quality must hold above our p95 script length, and here's the measured curve showing where it falls off."



Corn

What's our p95 script length, roughly?



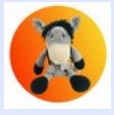
Herman

I don't know precisely, and I'd rather not put a guess in the document. I'd measure it. But it's long, and it's getting longer, and that's the point. The number on our spec is not the number on the model's spec.



Corn

Now the feature checklist, because that's the boring part of the document that saves you the most time.



Herman

It's a checklist and it should be a checklist. Context window large enough for your longest real input with headroom, so not exactly your longest input, but that plus room. Structured output enforcement. Tool calling. Image or audio input if you need it. Streaming.



Corn

Structured output enforcement is not the same as asking nicely for JSON.



Herman

It is not. One is a request in a prompt, the other is a constraint on the decoder. If your output contract is a schema, you need to know which one you're getting, because the first one fails occasionally and the second one fails structurally.



Corn

And the operational terms, which nobody puts in the document and everybody complains about later.



Herman

Rate limits in tokens per minute, measured against your peak, not your average. Uptime commitments. And the deprecation policy, which is the one that bites, because a model you've validated against can simply stop being available, and then your whole eval is historical.



Corn

Our workload depends on a model staying put.



Herman

Ours depends on it staying put and staying priced.



Corn

Okay. Multi-character support. This is the one I want to spend real time on, because it's where we're weakest and where the literature is best.



Herman

It's the interesting one, and the reason it's interesting is that it breaks the clean objective and subjective split. You'd think you could write a questionnaire and score it.



Corn

And someone did.



Herman

Someone did, and the result is the best argument I've seen for why a single quality score is a lie. It's a benchmark for long-horizon companion models, and it separates two things that everybody treats as one thing. Persona continuity, which is whether the model keeps following its assigned role, its boundaries, its values and its style. And trajectory recall, which is whether it can tell what actually happened in the conversation from things that plausibly could have.



Corn

So one is about staying in character and the other is about remembering the truth of the scene.



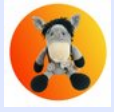
Herman

And the headline finding is that those two things rank models differently, and that's the piece that should change how we write our document.



Corn

Give me the split.



Herman

The highest scorer on persona retention scored zero point eight one. The lowest scored zero point five nine five. If you only looked at that, you'd pick the first model.



Corn

And the behavioural axes.



Herman

The model with the lowest retention score held all four behavioural axes ninety-nine point two percent of the time. The model with the highest retention score held them seventy-nine percent of the time.



Corn

So one model passes the questionnaire and drifts in conversation, and the other does the opposite.



Herman

The paper's own phrasing is that a companion can preserve structured checkpoint responses while showing more persona deviations in actual conversation.



Corn

That is our exact problem. That is the thing we've been describing for months without a name. One model sounds right in a ten-line test and by turn forty every character is the same character.



Herman

And if we'd only run the questionnaire, we'd have picked the wrong one and never known why.



Corn

So the qualitative requirement can't be one number. It has to be axes.



Herman

Axes. Role, boundaries, values, style, and measured across a long conversation, not a short one. And the length matters, because there's a separate finding that model-authored role-play quality degrades across turns specifically in naturalness and context maintenance.



Corn

Across turns. So the first twenty turns look fine and the problem is at turn eighty.



Herman

Which is our whole workload. We don't generate one exchange, we generate a whole episode with several voices in it.



Corn

And what about the memory of the scene.



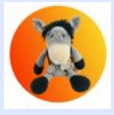
Herman

That's where it gets sobering. The trajectory results found user-state changes especially difficult, near chance in every condition. Chance there is twenty-five percent on a four-option question.



Corn

Near chance means the model is guessing.



Herman

The model is guessing about what state the user, or in our case the character, is in. And there's no context strategy that fixes it. They tested long context, hierarchical summaries, self-managed state, and retrieval, and pooled accuracy across all of them was in a narrow band, four hundred thirty to four hundred fifty-nine out of a thousand. They say it plainly. There is no universal context winner.



Corn

So summarising harder doesn't save you.



Herman

Summarising harder gives you a different set of failures, not fewer.



Corn

And the social pressure finding, because that one I find useful and slightly devious.



Herman

They found that emotional vulnerability, agreement-seeking, and realistic mixed schedules produced more boundary and style deviations than explicit adversarial attacks.



Corn

So the polite conversation breaks the model more than the hostile one.



Herman

The hostile one is what everyone tests for, and it's the easy case, because the model is on its guard. The realistic one is where it slips, because the pressured moment looks like a normal moment.



Corn

For us that's a character being gently pushed to do something out of character.



Herman

And that's a test case we'd never have written if we hadn't read that.



Corn

Now the judge, because this is the second-order consequence of all of it. If the qualitative requirement is that hard to pin down, who's grading it.



Herman

That turns out to be a variable too, and it's the one teams never document. They found that three different judges don't produce the same model ranking, especially on role and style. Same transcripts, different graders, different winner.



Corn

Which makes "we evaluated it" a much weaker sentence than it sounds.



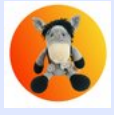
Herman

It's a weaker sentence than it sounds, and the fix is to report who graded.



Corn

There's a line about interns that I want in the document verbatim.



Herman

Treat the judge like an intern. An intern has no implicit context about your app, your users, or what good means in your domain. So you cannot write "score this one to five." You write "score one if the response cites the retrieved context, zero otherwise." An explicit instruction produces a useful score. A vague instruction produces a vague score that you then average and trust.



Corn

Which is worse than no score, because it looks like evidence.



Herman

And then there are the biases, which are known and controllable, and almost nobody controls them. Position bias, where the judge favors whichever answer appeared first. The fix is to run every pair twice with the order swapped and only count the verdicts that are consistent.



Corn

Verbosity.



Herman

Judges prefer longer answers. Which for a podcast workload is a disaster, because longer is not better, longer is longer.



Corn

And self-preference.



Herman

Don't judge a model with itself. It likes its own output. And before you trust any of it, validate the judge against human labels on fifty to a hundred items. If it doesn't agree with a human on examples you've labeled yourself, it isn't a judge yet, it's a random number generator with good manners.



Corn

So the document has a section that says who graded, what they were told, which model they are, and how we know they work.



Herman

Which is a strange thing to put in a model requirements document.



Corn

It's the bit that makes the rest of it true.

L

Larry

Are you tired of models that drift? Of characters who forget who they are by the third act? Introducing PersonaHold. PersonaHold is a clinically proven personality stabilizer for large language models, engineered to lock your cast in place for up to sixteen hours of continuous conversation. Simply administer one capsule of the requirements document into your deployment pipeline, and PersonaHold goes to work, reinforcing boundaries, defending values, and gently reminding your model that it is not, in fact, everyone at once. PersonaHold works while you sleep. Ask your evaluation engineer whether PersonaHold is right for you. Side effects may include unexpected verbosity, a sudden preference for whichever answer appears first, mild self-preference, judge drift, context window shrinkage, an inability to distinguish a user-state change from a plausible alternative, agreement with anything said in a confident tone, spontaneous summarization of conversations that never occurred, and in rare cases, the model will begin grading itself five out of five on all six AWS dimensions. Do not take PersonaHold if you have ever shipped on vibes. PersonaHold is not for everyone. PersonaHold.



Corn

Herman.



Herman

That's not mine.



Corn

That was read in the style of a man who has a capsule supplier.



Herman

I want to say formally that I have never described a document as clinically proven, and I've never used the phrase "gently reminding." I would remember. I have a spreadsheet of every phrase I've used on air, in alphabetical order, and "gently reminding" is not in it. Neither is "ask your evaluation engineer," and if you want I can pull the sheet up right now.



Corn

Don't.



Herman

I'm just saying the denial is available in a longer version.



Corn

So cost, and then I want to argue with you about the last piece.



Herman

Cost is a documentation problem, not a pricing-table problem.



Corn

Explain that, because most people treat it as the other thing.



Herman

Most people take the list price, multiply it by a guessed average length, multiply by requests per day, and get a number that bears no relationship to what they will actually pay. The advice is to compute from measured tokens. Pull the logs, get the real average input and output per request, and then multiply, including the retries and the tool loop iterations that the logs show actually happen.



Corn

Which is where the system prompt sits.



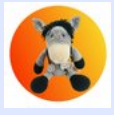
Herman

Being paid on every call, every retry, every loop.



Corn

And the tiering strategy. This one I think is better than it sounds.



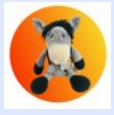
Herman

The advice is to test one small model, one mid model, and one frontier model, and specifically not to test three brands at the same tier. Because you're not looking for a winner, you're looking for the quality cliff. If the small and the mid both clear your bar, the cliff is below small, and you never needed the frontier model at all.



Corn

That's a different experiment than the one people run.



Herman

People run three frontier models against each other and then pay frontier prices for a job a small model does.



Corn

And AWS's worked example found no obvious winner across four models.



Herman

No obvious winner, so they tiered it. The most accurate but slowest and most expensive model for the premium tier, a cheaper one for the basic tier, a third for the free tier. Same workload, three answers, sorted by what the customer is paying for.



Corn

Now let me push on something, because this is where I think the episode's been a little too pleased with itself.



Herman

Go on.



Corn

Everything so far assumes you know what you want before you measure. And I don't think that's true. I think a lot of teams write the document after they've watched a model fail, and the failure is what tells them what to write down.



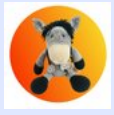
Herman

That's fair.



Corn

You didn't know persona drift was a way it goes wrong until you saw it. You didn't know the middle of the context degrades until you'd watched it.



Herman

No, I didn't. And you're right that the first version of the document is usually a post-mortem.



Corn

So the honest advice is not "write the document first." It's "write the document, and expect the first draft to be wrong, and expect to revise it the first time something breaks."



Herman

I'd accept that with one amendment. Write the document before you spend money, even if you write it badly. A bad document that's revised is still an eval. No document is vibes.



Corn

That's an acceptable amendment.



Herman

I'll take it.



Corn

Cost at scale, then, is the requirement we can actually hit, and context is the one we can't yet measure properly.



Herman

And multi-character support is the one that will humiliate us, because we just learned the questionnaire lies.



Corn

So if the answer to Daniel is "write it down," the real answer is "write it down, then find out you wrote the wrong thing, then write it better."



Herman

That's the answer.



Hilbert

You keep saying you write the document before you test. That's not how it happens. You write it after the first argument about it. The argument is the document.



Hilbert

I sat through one of these in March. Not a document. A form. Eleven pages, tabbed, with a section seven that was two pages long and titled "Conditions Under Which the Previous Section Does Not Apply." We had four meetings about section seven alone. Four meetings, nine people, eleven pages.



Corn

Nine people on a form.



Hilbert

There were ten, but one of them was there to take minutes, and I'm not counting her, because she wasn't contributing. She was writing. That's not the same function.



Corn

Fair.



Hilbert

The tab for section seven was a different color. Blue. Every other tab was manila. Somebody decided that and nobody could tell me who. I asked in the third meeting. Nobody knew. The blue tab was just there, and the section it marked was conditional, and the conditions were the only part anybody cared about.



Corn

So the argument at the meeting generated the document.



Herman

So the question of what your workload requires comes out of the argument, not the other way around.



Hilbert

The argument comes first. The document records what got settled. If you write the document first, you've written down what you assumed before anyone told you what you'd missed. And then you spend the meetings arguing about the wording instead of the requirement, because there's already a document on the table and everyone's defending their section.



Corn

That's a real cost.



Hilbert

Section seven didn't exist in the first draft. It was created in the second meeting because somebody said, out loud, "what happens when none of section six applies?" Nobody had written that down. It became two pages.



Corn

Did the document work?



Hilbert

Once it was signed, yes.



Herman

Signed by whom?



Hilbert

Eleven people. Nine on the form, plus the two who signed page eleven and never attended. They signed at the end, after the fourth meeting, and one of them signed a page that no longer existed by the time he signed it, because we'd renumbered the annexes twice. The signature was still accepted. Nobody checked the numbers.



Corn

Of course it was accepted.



Hilbert

I keep the blue tab. It's in a folder.



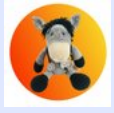
Corn

How long was it from the first meeting to the signature?



Hilbert

Six months. Six months and four days, if you count the day the second signer signed, which I do, because page eleven was dated.



Herman

And the requirement itself, the thing the form was for, did it change?



Hilbert

Three times. The first version of the requirement is not in the final document at all. It was in the first draft, and it was removed in the second meeting, and then it came back in the fourth meeting slightly different.



Corn

So the document forgot the original requirement and then found it again.



Hilbert

It came back different. That's what I said.



Corn

Where was this?



Hilbert

Doesn't matter.



Herman

So if Daniel's writing one of these, when does he write it?



Hilbert

He writes the first draft alone, badly, on purpose, to have something to argue about. Then he gets the argument, and then he writes the real one. If there's no argument, he wrote the wrong document.



Corn

The draft is bait.



Herman

I want to check one thing, because I've been thinking about the eval set sizing, and I think your page eleven is the same problem in a different coat.



Hilbert

I don't know what your problem is.



Herman

A document signed by people who didn't read the version that got signed. That's an eval graded by someone who doesn't know what good means. Same failure.



Hilbert

The signature is the signature. What's written above it is your problem.



Corn

Do you still have the form?



Hilbert

I have the blue tab.



Corn

Just the tab.



Hilbert

The form had to go back. The tab was mine, I took it off the ring before I handed it over. That was the right call and I'd make it again.



Herman

Do you still have the folder?



Hilbert

That's a separate question and I'm not answering it.



Corn

So if Daniel's going to write this thing, the honest sequence is a rough draft, an argument, and then the version that actually gets used, which is probably the one nobody wants to look at.



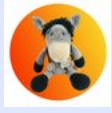
Herman

And the argument is where the missing requirement shows up.



Corn

The closing thought, and then we're done. If there's no canonical published template for a model requirements document, and there isn't one we could find, then every team is writing their own, and the quality of that document is doing more work than the model choice it's supposed to inform.



Herman

The closest things to a standard are the workload contract framing and the one-page spec. Neither is a spec, in the sense of a document you can download and fill in. So the thing that decides your evaluation, and therefore your model, is a document you invented, and its quality is entirely down to whether you argued about it.



Corn

The second thing to leave people with. If judge choice changes the ranking, and questionnaire retention and conversational behaviour disagree about which model is best, then "we evaluated it" is a much weaker claim than it sounds.



Herman

The document has to say who graded, and how the grader was validated. Otherwise it's vibes with extra steps and a spreadsheet.



Corn

The forward-looking shape of all this, as routing becomes normal, is that this document stops being a procurement artefact and becomes the routing policy. The thing that decides which request goes to the small model and which goes to the frontier one.



Herman

The document becomes the scheduler.



Corn

That's the version worth writing.



Herman

We should credit the man who keeps the blue tab. Our producer, Hilbert Flumingtop.



Corn

If this was your kind of episode, go back for episode twenty, Architectural AI; and episode three, Safetensors or something else. This has been My Weird Prompts. If you liked this one, a review helps more than you'd think.



Herman

You can send us your own prompt on Telegram at t dot me slash MWP listener bot.



Corn

See you soon.



Herman

See you.

SOURCES

What the research for this episode read before the script was written. Primary sources first.

1. **Anthropic (Claude) Platform Docs [primary]**
   Define success criteria and build evaluations
   https://platform.claude.com/docs/en/test-and-evaluate/develop-tests

2. **Salesforce AI Research [primary]**
   AnchorBench repo (ANCHOR long-horizon companion audit), created 2026-07-15
   https://github.com/SalesforceAIResearch/AnchorBench

3. **AWS Machine Learning Blog**
   Beyond vibes: How to properly select the right LLM for the right task, Claudio Mazzoni & Anubhav Sharma, 2025-10-17
   https://aws.amazon.com/blogs/machine-learning/beyond-vibes-how-to-properly-select-the-right-llm-for-the-right-task/

4. **Build with AI Today**
   Choosing the Right Model for Your Task
   https://buildwithaitoday.com/track/ai-landscape/learn/ail-choosing_right_model

5. **arXiv:2307.03172**
   Lost in the Middle: How Language Models Use Long Contexts, Liu et al., 2023
   https://arxiv.org/abs/2307.03172

6. **arXiv:2510.05381**
   Context Length Alone Hurts LLM Performance Despite Perfect Retrieval, 2025
   https://arxiv.org/html/2510.05381

7. **arXiv:2505.19293v2**
   100-LongBench: Are de facto Long-Context Benchmarks Literally Evaluating Long-Context Ability? (revised 2026-06-02)
   https://arxiv.org/abs/2505.19293v2

8. **arXiv:2512.04307**
   Evaluating Long-Context Reasoning in LLM-Based WebAgents, 2025-12-03
   https://arxiv.org/abs/2512.04307v1

9. **arXiv:2509.17694v2**
Evaluating LLM-Generated Versus Human-Authored Responses in Role-Play Dialogues
https://arxiv.org/html/2509.17694v2

10. **Chrome for Developers**
Design your evaluations / Your mental model for AI testing
https://developer.chrome.com/docs/ai/evals/design

11. **LangChain**
evaluator-design reference (LLM-as-judge vs code decision framework)
https://github.com/langchain-ai/langchain-skills/blob/main/config/skills/langsmith-online-eval-engineering/references/evaluator-design.md

12. **Arize AX Docs**
Evaluator Best Practices
https://arize.com/docs/ax/concepts/evaluators/evaluator-best-practices

13. **aaai.latere.ai**
Choosing a Model (workload contract framing)
https://aaai.latere.ai/en/practice/choosing-a-model

14. **Microsoft Learn**
Choose the Right AI Model for Your Workload
https://learn.microsoft.com/en-us/azure/architecture/ai-ml/guide/choose-ai-model