Probabilistic Nature of MCP

31 MIN · YOUTUBE · HTTPS://WWW.YOUTUBE.COM/WATCH?V=ND1CANARG-K
https://www.youtube.com/watch?v=nd1CaNArG-k

SUMMARY

Broth Mesh, CEO of MCP Jam, discusses the importance of understanding the root problems in development rather than jumping to assumed solutions, a concept known as the XY problem. He emphasizes the need for developers to evaluate what they cannot see in user interactions with their MCP servers, advocating for better observability and user data collection to enhance user satisfaction and tool effectiveness.

- *The XY problem occurs when developers focus on assumed solutions (Y) instead of the actual problems (X).*
- *Understanding user intent and context is crucial, as many issues arise from invisible interactions between users and agents in host applications.*
- *Developers often only instrument the tool call portion of the user value chain, missing insights from user interactions and agent reasoning.*
- *Effective MCP development requires real user data to inform evaluations and improve tool performance.*
- *Strategies for gathering user insights include adding parameters to tools, utilizing existing AI product data, and creating isolated test environments or sandboxes.*
- *The development process should incorporate a feedback loop (flywheel) that captures user data, generates insights, and iterates on tool quality.*
- *MCP Jam is launching tools to facilitate better testing and feedback collection for MCP applications, enhancing the development process.*

All right, thanks for coming. My name is Broth Mesh. I'm the CEO of MCP Jam. And uh this is Marcelo, our CTO. Big hand for Marcelo. Nice. Appreciate you. Uh we're going to talk about something that likely many of you aren't doing yet. Um yeah, let's talk about it. Evaluate what you can't see. Actually, I'm going to get a quick show of hands. Uh can you raise your hand if you're an MP server developer or work on the MC server development team?

Okay. Okay. Most of you guys. Uh how about are you on the MP host application side in any way? Okay. Great. A few of you. Few of you. Um managers. Managers. Tech leads. Great. Great. Um do you lead AI strategy at your company at all? Sweet. Look at that. I saw a few hands there. Nice. All right. I just want to get a sense of things. Uh, this is what I posted on LinkedIn and uh, it it sounds pretty pretty uh, catchy, kind of gimmicky, too. Um, raise your hands if you're familiar with the XY problem.

Oh, okay. That's nice, actually. Uh, so let's start here. So, if you're a developer, you might know about the XY problem, but I'm surprised that only a few of you do. Um, and that's great. I love the XY problem. I think a lot of my conversations kind of boil down to to this problem specifically, whether it's workrelated or personal related. I'm going to break it down for you. So the XY problem happens when someone is asking about their assumed

solution to their problem instead of asking about their problem. So you can imagine I'm like a platform engineer. I have like a junior developer coming up to me and they're like, "Hey, um, how do I improve our remote jobs async framework to better optimize this specific use case by 2%." And I go, "Okay, great. Why?"

So, if you ask why, usually after a few or maybe hopefully after one Y, you get down to the root of their problem. And their problem is the X. Um, and the XY problem happens a lot in that specific scenario. That's a that's a good example there. um and it can waste time. Um so when you're focused on the Y, the assumed solution instead of the X, also the solution space gets narrow as well. I'm gonna go through some examples just to really hone it in here. I'm going to call back to this quite often. Uh so bear with me on this. Okay, so went through the definition here, right?

We're good here. Let's go through an example. All right, how do I breed a faster horse? Okay, that's that's the question. That's the assumed solution from this person. You know, that's going to happen to you very often in your life, so we just got to walk through it together, right? Um the X. So, after some conversation, you kind of get to the point where you realize this person actually just wanted to get their kids from point A to point B faster. And it's like, okay, if that's your problem, then skip there. If that's your problem, then there were probably quite a few options there that you could decide on to get from point A to point B faster instead of breathing a faster horse, right?

Public transit, bus, train, you could get a car, right? So, it's not clear from the original question um what to do from there if you're focused on the why. But as soon as you get down to the yets, now there's a lot more you could talk about. Another example, um MCB sucks. It eats so much context window. How do I build a CLI? I was forced to throw this in here. I am not in this whole MCP CLI skills debate camp. I am off of social media. I'm glad to be honestly. Um, but I did figure I would dive into this. So, what's really happening here is the problem is agent workflows are inefficient for this person. Um, and what that means is, okay, their costs are ballooning when they're connected to several MCP servers. There's a lot that we can do when we focus in on this problem instead of focus in focusing on the solution. Now, we can go into arguments about MCP versus CLI versus skills and that's just not worth the time. Um hopefully for this group of people in this room, but let's go into the solution space, right? When you go into the solution space, then you have a better sense that hey, there's actually a lot that specifically an MCP host application developer especially can do to resolve this. And this was mentioned in the keynote as well. Like one of those is, hey, what if we figure out some form of progressive disclosure for tools, right? Instead of when you connect to an MTP server, have all of the tool definitions and manifest data thrown into the context window for your workflow. Hey, a little bit better is probably what if if a user goes through um some common workflows and you have an idea of that user hitting two or three tools all the time, maybe throw those into the tool manifest cache and leave the rest out of it and only semantically search for those tools after uh a prompt. Right? So, there's a few things there. I'm just going to throw that there, but hopefully this clarifies some of the things that I'm going to talk about later. Um, and it gets down to the ats and wise of what I think MCP um, going into quite often.

And I think there's a reason for this. And you're probably thinking like, okay, I'm talking a lot about X's and Y's and this whole talk was about evaluating. And raise your hand if you thought this talk was about evaluations,

eval testing. Oh, okay. Nice. Nice. Okay, only half of you. That's great. Um so the reason MC teams I think are more focused on the why is because this ecosystem is fast moving. There is some real user and business opportunity here that needs to be extracted and the first mover advantage is is very real. Um because the MC ecosystem moves quickly now all these new hosts host capabilities SDKs protocol features feels like there's always a new thing to stay on top of and because the real user and business opportunity is there you want to stay on top of it and make sure that you're extracting as much of this opportunity as possible. There's also a problem. And the problem is this system is really hard to observe. And because this system is really hard to observe, it makes it even more likely that we're going to latch on to the first solutions that come out there that are popular.

Um, yeah. So, I talked a lot, but like who am I really? Just like I just said, I'm like the CEO of MSP Jam. Have any of you use MPJ by the way? Oh, okay. This is great. This is awesome. Okay, so these are some peeps from MCBJM that are here at the summit. It's three of us. Feel free to say hi to us afterwards. Um, but here's who I am. So, I was a former tech lead at the sauna and I've been there for just over four years. I've been on the API developer platform team there the entire time while I was there and over the last year or so I was the tech lead for them.

So, I owned our public rest API or OTH authorization server and our MSP server platform. Um, and you can imagine I've seen a lot over the last year, um, from an enterprise point of view over there. Um, my onboarding to MCP came during the security incident, a very public one. Um, and that was when I had to figure out that, hey, the thing that we built really quickly for a launch moment with Enthropic was um, not great in a bunch of different ways. Diving into telemetry essentially figuring out what do we know and what do we not know? And that was my intro to MCT. That was this time last year essentially. Um afterwards I around the time the apps SDK released um I prototyped the MCP app for chat GBT for ASA and then led a group of tiger teams on several MCP initiatives to ship our chatbt app to launch our cloud MCP app and to run an MCP off migration from closed DCR or sorry open DCR to closed DCR. This was around last year. And then finally from close ECR to pre-registration up until those launch moments. So I kind of see a lot of MCP from Asauna enterprise angle. Um oh and recently I also proposed a sauna being an MCP host application uh as so many other applications are doing right now.

Um so yeah that's me. Oh, also I was a champion for MCP Jam while I was at the sauna and it was really useful for us to test debug MC apps and our server, our T server um across host applications, host capabilities. Really awesome. Good stuff. Okay, so I'm going to dive into the problem and I'm curious after I go through the problem if you guys are actually hitting these problems just for some validation. So, here are two sessions from your MCP server logs. And let's just assume these are tools that you expose in your MCP server. Uh, left side, right side. You see both sides.

Sorry, it's kind of hard to see for people in the back, by the way. Both sides, uh, you have a user in one session around the same time stamps for both of these users. This session, um, let's assume they're in the same host application as well. And get project status is called twice. And that's what you see on both sides. you're like, "Okay, great. Both return 200 status codes. This is awesome. Latency looks fine." Um, different users, different sessions, but very similar looking sessions. Um, from your server's perspective, these look pretty much identical,

right? Two successful tool calls, everything's green. You see four get project status tool calls added to your count of tool calls for the day, right? And let's dive in. Um, I'm highlighting the stuff that I just mentioned.

And then let's look at the the left one and just look behind the scenes of what's actually happening in the host application. Yeah. So, if you look behind the scenes, here's what's happening. All right, cool. Um, so the user wanted to fetch their Sprint project and then got back a response. They were happy with it. They're okay. Okay, great. Let me get the book club project status 2. Okay, great. Um, now give me an outline. And they got value from those two tool calls from the app.

And now they're moving forward with their lives. Pretty sweet. User satisfaction is awesome. Let's move forward. What about the right side? The right side, same tool calls, but this time the user is very unsatisfied with the responses. So they wanted to fetch some product launch project. They got back a game dash launch project and that's not what they were looking for. Then they tried to prompt again. This time they got a channel partner looking project. It just wasn't the one they were looking for and then they were frustrated. Right? So same logs, same server, completely different user outcomes, user satisfaction on either side of the spectrum.

So why is this a problem in MCP? Um, so your user isn't in your application anymore or doesn't prefer to be in your application anymore. they prefer talking to an agent and that agent is talking to you or your application. Um the user and agent live inside a host application. Let's say chat evbt claude claud coowork cursor um notion and your server lives outside of that service boundary. Right?

So that dotted line is the system boundary. Everything to the left of it that's where the user actually prompted. That's where the agent actually interpreted the user's intent and the context behind that user. It decided how to choose your tool. it decided how to parse your response. Um, and you can't see any of it by default. So, you only see what hits your server. So, the user's intent, the agent's reasoning, the selection decision, all of that is invisible to you by default. So, this is by design. MCP is a protocol meant to return tools to agents. Um, that agent has cannibalized all of the user contents and intent and you see none of it. Um, and when you don't get called, whether you were even selected is also missed, right? Um, and you never see the prompts that you should have handled.

Uh, this is an aside, a raise your hand if you have built an MCP app. Wow. Okay. Oh, okay. Few few uh slight hands. I don't know what slight hands means. You're like halfway there to an MP app. um would recommend and I think I would recommend for the new user acquisition flow um opportunities here. So here you see I'm using Figma in cloud and Figma in claude is pretty awesome just because I can create a diagram anytime I want without actually being logged in. And what's pretty sweet about this is let's say I am looking to just create diagrams period in claude. If I get discovered or if I get recommended Figma to go use, I might go use it and then decide, wow, this is pretty awesome and I have now hit an aha moment with Figma before I even land in Figma. Um, so we might talk about this later, but yeah, wanted to throw that out there.

Uh, a little bit more about the MC value proposition actually. So here this is actually me in cursor yesterday and I just decided to you know um I think this is using glass maybe. Yeah I'm I've opened a browser there on the side

and I'm like you know prompting the agent and something interesting shows up and this is some new discovery features for cursor now that they have MV app support. Um, and you can see when I when I said how do I track something, it actually recommended linear to me. And this is pretty sweet because if a host application is now letting you be discovered, you as an NP server developer be discovered. Um, this is a pretty sweet user value proposition. Um, then let's keep going.

So, let's talk about the user value chain. And this is just something that I don't know kind of came up with. Um, it's a framework. Frameworks are fun. Frameworks are fun. So, there are six steps for how a user actually gets to some sort of user value from you. Um, and how they get to some sort of user value from you is one, they have to get connected, right? Um, so they first have to like go through a consent screen for the most part. I mean there are some mist tools there are some tools or some servers that don't require O but you need to find some way to get them to be connected um and does the actual tools list call from the host application to you succeed then does the agent know that you actually exist right um did it even find your tool pick your tool kind of a big deal um and then once they actually discovered you did your did the right tool get chosen by your MCP server Um, cool. Let's keep going. Let's just say that they did call your tool correctly, the right tool, but did they pass the right arguments for that tool, right? What about the right schema for those arguments? All right, let's keep it going. Let's just assume yes for all these things. Um, tool response. Okay, your server delivered a great response for the tool call presented. Uh, you have great endto-end testing, unit test, you know, all the stuff that you do for your server is correct deterministically. Um, but did the agent actually reason about your response, figure out the right information from that response to go give user value to the user? Who knows? And then finally, did the user actually get value from your MCP server? That's a list of a lot of steps to go through for a user to actually get value from your MCP server.

But those are the steps. And if we walk through some of these steps, I'm going to go back a bit actually. in this chain here in the beginning I circled tool call that's what you actually have full control and observability over um if you're an MCB server developer you are instrumenting that portion um so at every step in that chain you are potentially losing ground and dropping off user value, but you are only instrumenting that part right there potentially. And we can talk about maybe some ways that maybe you are instrumenting the user value chain in in interesting ways. Um, but I would guess the vast majority of you are only instrumenting that portion right there.

Let's keep going. All right. Uh, this is like kind of important, kind of an aside, but I'm an API guy. I've been on the API team for quite some time. Um, API rest API design is really interesting because you can kind of think of MCP as being like somewhat directionally similar right to REST. Um, but who you're targeting is actually very different, right? Like for for MCP, you are literally building for an agent.

And that is very different from building for human developers. And I mean, I say human, I just feel like I had to throw that in there. But an app developer, what they care about is the breadth of your REST API, right? Like they want to make sure if if the API is not a core product for your application, um they want to make sure that your API has all the functionality that your web application does, for example, right?

They want to make sure that for every endpoint that you expose, you have tons of parameters to let them configure everything that they need from your API and get data from your service. Awesome. And that's what you do. You design in a way that's ergonomic for a human developer. And a lot of times ergonomic for a human developer is, hey, um, just give me all the features that I want so I can pick and choose what I actually want to build for my integration. Right?

Right. So, this this is awesome for an app developer because now the app developer can decide here are all the awesome integration features that I want to build out. Um, that's very different for agents. Um, I'm going to go into a quick story about evals and wow, I'm really running out of time here. Um, I built our first evals and what happened when we built our first evals. Raise your hand if you know what eval are.

Beautiful. Raise your hand if you're instrumenting your agents using evals. Oh, raise your hand if you're instrumenting your SP server using evals. Okay, gotcha. Wow. So, lots of people know about evals aren't actually instrumenting with evals. Um, I built our first eval out of the sauna. And when I built our first evals out of the sauna, we got into a room, my PMs decided, here are the workflows that I think users are using that we should instrument towards. Uh, I made a golden prompt set. They're they're called golden prompt sets. They tell you here are the workflows that users actually probably care about doing with your MCP server. Come up with those somehow and then decide to test against them. Like here are the set of user prompts that we think people care about to fetch projects for example direct indirect negative and then actually run through these. Evals are non-deterministic tests. They say given a user prompt uh did my server actually deliver value for that user prompt in the ways that we expect. Did the right tool get called and the right arguments? Right? Um that's all based on something. It's all based on incomplete data because the system itself gives you incomplete information. And I'm going to skip through a bunch of stuff here because I'm running low on time. But here's how to actually get user contents in hopefully really direct ways. One, what if you just add a parameter to all of your tools? You're talking to an agent.

That agent could just give you all the user contents if you ask for it, right? You could say, "Hey, by the way, what's the user intent?" When you pass in a tool or when you pass in arguments for my tool, just like for every tool, give me the user intent. Give me maybe the semantics of how the user or sorry, not semantics, uh you guys know what I'm talking about, the the feeling of the user when they're passing in uh this content. So that second user might be like, "Hey, I'm kind of frustrated that the first two tool calls didn't work for me." So you get that that meaning across. Um, so that's cool. There's some downsides here. Of course, some downsides are now the big host applications have an app review process. They will actually look at your tools and say, "Nope, we're not going to allow that. It doesn't actually benefit the core value proposition behind this tool, so let's not do that."

It's essentially stealing contests from the host application. They want to keep that probably. Um, another reason is this affects the usability of your tool because there's a small chance now that now your tool is, I don't know, between five and 50% less effective because now the agent has to reason about giving you user intent alongside the core arguments behind your tool. I have no numbers to back that up. You can get numbers though if you run evals. Um, another one, what if you use your existing AI product data? you probably have a chat bot somewhere and if you have the sense of user prompts, you should actually use those user prompts and actually instrument

towards those. It's it's a good first start, right? When I first built my first eval set, I basically took all of our internal uh chatbot data. I got some guides that told me here's how to improve my tool sets. And I ran it through deep research across all the host applications and got into a room with my PM and said, "Okay, cool.

This is right." Right? And then that's the eval test or test that we run every time on CI. Um, but yeah, this helps a lot actually. It's not perfect and it's not perfect just because uh your existing users are probably power users. They know how to use your product. They're in a surface where they can probably see something about your product. That's very different from a user in chat GBT who maybe doesn't even know about you, right? So, not a onetoone comparison, but at least it's something. What else? What if you had a test environment and that test environment was basically a host application where you could collect feedback? So you host this. It renders MCP apps really well. Maybe uh you can connect to multiple MCP servers there, but you really care about instrumenting your MCP server. So you say, "Hey, I I want to QA this." You send this environment to your team. You say, "Hey, go test out prompts against the MCP server." Well, if you dog food in your company, you can test out prompts in this test environment. Um, you can even share it to beta users, your potential ICPs, your ideal customer personas, um, and have them use it. Now you have really rich data from the set of users that you've shared this environment with that you can get insights from and actually build out evals for. So, pretty sweet. It is offline though. By offline, I mean it's not production traffic. It is you sending this environment to someone, right? But still pretty awesome.

Uh the meta field I'm gonna skip. It's if you aren't familiar with the meta field would recommend. I'm running very low on time so I'm just going to keep it going. Sampling. You're like, "Hey, but uh breath mesh. Isn't there a feature in the MCP protocol that lets you actually get data get contexts from the agent on the other side?" Yeah, it's called sampling. Um 99% of host applications do not support sampling. And why is that? Um there's that's where like there's an idiom. It's like the something hits the road like it's not practical for most applications, host applications to support sampling. And I say this because they have data imperative to keep all the data to themselves. Why would they share that with you, right? And I don't know, maybe that's a hot take or something, but I don't know. Seems pretty straightforward to me. They probably want to monetize off of that.

build an ad platform around it. That's why 99% of them don't support sampling. I don't know, maybe there's some host application developers here that will prove me wrong. But yeah, and then MC apps I think are awesome. Not just for that user value proposition, and if I'm running over time, please just stop me. um not just for the user value proposition, but in that aha moment, getting new users to discover you, see your brand, see your reputation, and actually play around with a pretty awesome app. Um, but also because you can instrument it. And this is where the user actually interacts with you instead of you interacting with the agent. And that's pretty awesome because on every button click, on every interaction within that user interface that lives in that sandbox iframe of the host application, you can instrument that. You can get all that metadata and figure out, hey, here's what the user actually wants to do with my UI. Wow, this is pretty awesome. So, if you aren't doing this yet, would recommend for both of those reasons.

Cool. And then there's a flywheel. I'm going to rush through this so fast, but this is pretty important. the flywheel. You can see evals is a part of that. You probably a few of you came in assuming this talk was about

evals. This talk is about this flywheel and how to make effective MCP. And effective MCP starts with real user data. And I gave you six ways to get real user data right now. And if you aren't getting it right now, I would highly recommend starting there. Um, and then you capture that signal. You turn it into insights. You cluster, right?

You take user prompts. How are they actually using your MCP server? Take that real data, cluster it, get workflows out of them, get insights out of them, and then eventually build an eval test framework on real user data, not just makeelieve imagined workflows. Eventually create a quality gate. You should not ship unless you hit your quality bar. Build it into CI/CD. After that, ship with confidence. See how users actually interact with it. generate the flywheel, actually get it spinning. Right now, it's stuck for probably all of you given the hands that were raised that said they didn't have evals yet. And we got to start spinning this flywheel because MCP servers, getting it out there, building and shipping is only half the cycle. The other half is actually delivering user value.

I'm going to keep it going because I know I'm running out of time. I'm surprised no one has stopped me yet. I'm I guess I guess I'm just going to keep going. Um, sandboxes are pretty sweet. Why they're also pretty sweet, this is going back to the isolated test environments, is uh it gives you a really safe place to QA and dog food and get user insights in an environment that is outside of the host applications, chat, GBT, Claude, etc. Um, and what's really awesome about this is, uh, raise your hands if you work for an enterprise.

Okay, great. Raise your hands if you can dog food your MCP server internally. Okay, about half of those hands went up. Um, I'm going to give you one reason why you might be blocked from doing that. When you are an enterprise company and your MCP server is distributed outside of your company, um, if you're on ChatG and Claude, your personal accounts, um, if you're on ChatG and Claw with your personal accounts, they can train off of any of your data, right? So if you happen to be connected to your enterprise MC MCP server for enterprise workspace accounts, they can train off of all that data. That's a huge data at filtration risk. So a lot of security and data privacy folks will block you from actually connecting to your own MCP server at your enterprise. That's a huge deal. Sandboxes, test environments fix that um because by default they won't train on your data.

Operationalize this. I'm still surprised no one has stopped me from speaking. This is great. Um, how much time do we have? >> Great. Love that. Seems like it. This is awesome. Um, here's some goodies. I'm from MP Jam. Yeah, go ahead. >> Love that. You're too kind to me, sir. Way too kind to me. Let's see. Cool. Can I click this? I can click this. We're showing some stuff. This MCB jam uh it's an inspector and it is much better than the open source inspector that you might have used already. Uh because it actually lets you test MCP apps and renders them with both support for open apps SDK, MCP app standard, um test across host capabilities. This is sandboxes. Those isolated test environments I was talking about. We're launching next week and I'm really excited about some of the products that we're about to build. This is essentially your own host environment that you can lock down that you can share with anyone you want. Set the system prompt, the temperature, actually test your MCP servers in chat in the sandbox, share it with faults. You can say, "Hey, go test your MCP server in my sandbox. This is pretty awesome."

Eventually, maybe I can skip through some of this stuff, but you can get into you. You shared a link with someone, they are now testing your MCP server in a sandbox environment that renders MCP apps, which is pretty awesome. And what if in this same isolated test environment, you can ask for user feedback just straight up, hey, one through five, did you actually enjoy your interaction here? Um, give me open-ended feedback on my MCP server as well.

Uh, and then look at the insights. You can see the full trace of what the users actually did and the feedback which is pretty sweet. Uh and then dive into this trace here and you can actually see there's a trace all the way at the end. Learning platform learning is is pretty good. But yeah, you can see this trace and this trace is pretty sweet because it actually breaks down, hey, here's where the tool actually went through for the user, for the agent, and for your MCP server. You can see exactly where the latency spent time for your MCP server in this case really quickly. But the user actually felt a little 5-second difference there because the model took that much time. And that's really important. You can click on that trace, generate evals that you can run in CI/CD just like that. And that's pretty sweet.

I took up way more than your minute. You're too kind to me, sir. And if you want to stay connected, hey, I'm here. >> [applause]