

[00:00:01,000 - 00:01:12,000] What up, fam? I'm here the Sundial Bridge in Reading, California. And today, we're going to have a conversation about sort of what I think the future of good market engineering looks like and in some ways the future of software, especially for folks that have to build things that are the same for a living. A lot of what I do is the same. It's different data sets, different remixes of the same thing, but the messaging framework is pretty much the same. The way in which I query the data is pretty

[00:00:38,000 - 00:01:49,000] much the same. The types of things I need to do are all pretty much the same. So, you'd be kind of surprised how the convergence of good looks the same. But the thing about it is that our tools still don't keep up. There's no compound interest in our tools today. If I build a workflow, I got to build that same damn workflow again. I got to go find it. I'm not an organized guy. But what's changing about that is that my intelligence can be strewn everywhere. So everything I've ever

[00:01:18,000 - 00:02:49,000] done, everyone you've ever known, every exist on the pale blue dot. It's a great speech by Carl Sean. Um or writing actually. So the thing about software and code and things like cloud code that make everything different is our context is saved. It knows who we are. It knows what we've done. It knows how we've done it. It knows what doesn't work. The context compounds and kind of imagine that, you know, we've had a world where, you know, you think about email or notes or all these note-taker apps, right?

[00:02:06,000 - 00:03:12,000] The idea is that your context over time will help you do all sorts of things from update the CRM. Um, you know, you'll see this in apps like Fixer, which do a pretty good job at like drafting emails for you. I just canceled my subscription, though. Uh, and that's because context compounds. The more context the models have and the better they're getting at sorting, it just gets more and more valuable. Well, what does this mean for you? Well, it means that you should probably start trying to capture your context in

[00:02:39,000 - 00:03:53,000] code as quick as possible. You should try to do more of your things where your context can compound. And if your context is trapped in tools that don't talk to the large language models, it's really hard to translate that because that's stuck in some UI and UX somewhere and it's a real problem for your future productivity. I have um just closed more deals than I've ever done, which is going to be a problem because it's mostly just me. Oh, and my good friend Nikki, too. Um but

[00:03:18,000 - 00:04:33,000] but I'm getting better and better at doing at shipping campaigns quickly, good campaigns with opinions about messaging. And every time that my client does QA, they add more and more context. And every time I get a new client, I can take all of the learnings from the past client, both in terms of the code and the context I gathered to build a new context engine. And so I think that if I were starting right now as a go to market engineer or even a product builder, I would think about this compounding context that you

[00:03:56,000 - 00:05:07,000] can talk to as a differentiator because every time I get into a clay table, it's day one for me, which is kind of dangerous for Clay. They're they're working on They're an incredible team. But if it's day one, that means that the way in which they compete as a goto market kind of the deacto go to market engineering tool is with more integrations and better integrations and better workflows and thoughtful user experience. And those are great competitive advantage and the community

[00:04:32,000 - 00:05:44,000] great competitive advantage. But but a competitive con a competitive advantage where um Claude or another tool can talk to all of that context and empower me as the user and there's going to be more frontiers like this that is a durable competitive advantage because why am I going to switch to another tool that's going to silo all of my efforts? And so I think that and I also think that one of the things that I wish that I didn't have to build again, but I just don't want to integrate with

[00:05:07,000 - 00:06:17,000] proprietary software as much anymore is there are some things that the clay agent is exceptional. There are some tools in clay that were just thought of so that were just done so well. But I'm really trying to I I just have to the business forces uh uh for me to capture more revenue and to provide more value to my clients. I have to be able to transfer my context to them. And in the past, that's been teaching them tools. But I don't want to do that anymore. I don't want to teach

[00:05:43,000 - 00:07:01,000] them tools. I want to teach them how to think, how to talk. And my hope is that they can talk to all of the context that we built up that they can start, you know, and and by the way, connected to databases, which none of these tools really do, um, where it can talk to data bricks or or even a bunch of CSVs, gigabytes of CSVs. Those are all campaign potential. And so this is kind of what I'm angling towards which is almost all of my work is going to be done in cloud code. Uh it's almost all of it is going to be

[00:06:22,000 - 00:07:35,000] well actually maybe all of it is going to be checked in to a repo provided to my client. I'm going to teach them how to talk to launch a campaign. I will hand over some Claude skills, messaging skills, and other skills that I built that teach that basically I've taught the model I'm still working on my posture. See, see, I [Â __Â] up. Um, uh, that I've taught the model basically how to think like me and you can invoke different parts of what I believe in your build. And so, so this is sort of a note to all future

[00:06:59,000 - 00:08:07,000] software providers. Think about this. Think about how you give your users compounding context. Think about how you tie in with these other coding tools. Um, allow the coding tools to do the most popular workflows in a iterative way, which is like go launch this thing, check it, and come back. And I think the software companies that build these really great loops between what what's my intention, run the loop, um, and validate the loop, there going to be a lot of value there. Yeah. So I I want people to think about this as

[00:07:34,000 - 00:08:49,000] they build. How does your com text compound? How do you deliver that compound value to your clients? How do you in your day-to-day work save what you're doing and this will all get way faster. I was teaching my friend claude code the other day and he just popped in a new cla code thing and he said hey let's continue it's like continue what right it's stateless um so it doesn't know but that's why you need a process to capture just the right amount of context and you do need to at

[00:08:12,000 - 00:09:16,000] least today you need to filter the world and you need templates and heruristics to filter the world as much as I hate it some of that is ICP persona uh the call recordings, the segments, you you need to parse the model. You need to basically give it your intuition and and how you're thinking about the problem and also what you're attempting to do. I saw someone that wrote a post on LinkedIn the other day and they were like, "If you need to be taught posture, if you need to be taught AI, like

[00:08:44,000 - 00:09:52,000] there's nothing to do, you know, it's like it was some negative post." And my kind of thought on it was that we have a machine that you can ask any question of. And so a question of how do I learn this thing is usually just getting started and asking the thing how you learn the thing. We've never had that before. It's been obnoxious to learn things in the past. You know, you got to get a book and you got 12 pages of the 200 are useful to you. But we now have a way to talk with all

[00:09:18,000 - 00:10:24,000] information. And if that's the case, we need to think through, well, if I had all information, how do where do I begin? Well, this is why it's important for the model to be able to connect to your systems, connect to your tools, but most importantly, only have the context you care about because then you can build a world of context for you. And the more projects you do that have your context, the more it knows what you're doing. So, it's my thought of the day. Have a beautiful Saturday.