# [Transcript] FYI - For Your Innovation / Orchestrating Enterprises with Data and AI with Palantir CTO Shyam Sankar

Welcome to FYI, the four-year innovation podcast. This show offers an intellectual discussion on technologically enabled disruption because investing in innovation starts with understanding it. To learn more, visit arc-invest.com.

Hello, everybody, and welcome to FYI, Arc's four-year innovation podcast. We're super happy to welcome Shyam Sankar, the CTO of Palantir on the show today. Shyam, thanks for joining.

Hey, thanks for having me.

I think it'd be great to start off just to give our audience, our listeners, a little bit of your background and how you really found Palantir and how you've seen the company change over time because I think if my research is right, you were the 13th employee at the company and have been there for quite some time, so I'm sure you've seen a lot over the company's life.

Yeah, I joined Palantir 18 years ago, roughly as the 13th employee and really the first person who built out the team that subsequently became known as the forward-employed engineering team. So working with customers in the field to really almost solve through back propagation what the product ought to be based on the empirical truth of what the problems were that they were facing and the answers that they needed. But I came to Palantir because of the mission. I grew up as a child, essentially, of refugees. We fled violence in Nigeria, and I was raised with the immense gratitude of understanding the counterfactual every single day of how lucky I was to be in this country. My uncle was a victim of the 13-coordinated train bombings in Mumbai in 2006, and so I could think of nothing more impactful and important than working in problems related to national security in some way, shape, or form. Of course, the business has grown to be something much bigger than that, but I think that the fundamental driving ambition of solving problems that matter, understanding that the technology we're building has an impact on society, so you should own that and think about that in a normative way, has always been there as we scale the business from government to commercial and beyond.

I was just going to ask quickly, Frank, Shom, I remember first hearing about Palantir, and the way it was described to me or us at our commitment might have been my prior life, but it almost sounded consultant-like to me, and then I looked at your gross margins and I said, this is no consultant, but I still think that misperception is there. I don't know if you find that when people are trying to explain what Palantir is and what you do. Can you talk about that misperception and how you are able to help people understand what exactly you're doing?

Yeah, it's a great question and certainly I think one of the most common misperceptions of the company, and I think a lot of it is how we present ourselves in terms of our product development strategy. Now the term is much more often copied by other companies for deployed engineering, but the concept of that was really like, what is the likelihood that sitting in Palo Alto, eating blueberries, we're going to think of the exact problem that our customers have in

these very specific domains, like what is our access to truth? Is it here in headquarters
or is it out there in the field? And what is our product development methodology going to be
as a consequence of that? And our idea was that we actually need to go walk not just a mile,
but many miles in the shoes of our customers to understand not only what their problems are,
but the deeper secrets that in some sense their current problem is symptomatic of,
and then go solve that root issue. And so there's a process of both discovery
and product development in doing so. And I think the conceptual metaphor you can think about that
is actually how do you train a neural network? You start with the answer and you back propagate
to train the function. So similarly, how do we develop our product? We go figure out what are the
deep fundamental unsolved problems that exist out there in the field and then work backwards to
the
solutions around that. And I think that's also given a sustainable edge in how we do R&D.
You can kind of think about it as the set of things that are discoverable sitting comfortably
in headquarters have been discovered. But as fancy as your ERP system is, when I go to the
front lines of these factory floors, they're running on Excel. So if ERP system works, why are they
running on Excel? And who's spending their time thinking about that question? And can you go
solve that? Can you build the product that addresses those issues? And so absolutely,
I guess this is a long way of saying we can present like a consulting company, but we're not
billing for hours. The model is not actually services. It's the model is to go figure out
the profound problems that affect our customers. And you can see that in the early way in which
we went to market, which was like really people would call us with a problem to solve. Tom Enders
needed to fix the A350 ramp in the initial production, right? He's not a technology buyer. He
has a really profound business problem that needs to be solved. How do I get my ramp back on
track?
I was supposed to produce 50 planes in my first year of production. I'm on track to produce 16.
I've got six months left to go in the year. And certainly, now you have to go say, well, how do
I go sit at the final assembly line and to loose? How do I go understand what's going on and send
a zero? How do I pull this all together and understand the parts of this that are technical,
that involve data and not flowing between institutions? How do I think about this institution
as a web of decisions? From the hand of their supplier to the hand of their customer,
they're making a bunch of decisions. Those decisions are all connected together.
We've organized as a society. We think of these companies. We've broken them down functionally.
This is your function. That's your function. Those functional divisions are a manifestation
of how we're dealing with complexity. Well, I know one person could make all these decisions.
But actually, software can really help you do that. It can help you understand when I'm making
one decision close to my supplier, how does it affect the customer? When I make these decisions,
what options am I losing? What options am I gaining in terms of how I can deal with this problem?
To make that less abstract, kind of a simple example that's real,
if my procurement person is incentivized based on the KPI of raw material costs,
and my production person is incentivized on yield, you have a problem where the procurement
person might be buying the cheapest possible good that has a lower net yield than a more
expensive raw material. And this is this sort of canonical problem. If you play it out of a more
complicated value chain, where it's not only that the left hand doesn't know what the right

hand's doing, the left hand is actively hurting the right hand. And so how do you connect these decisions up so that your company can be better? Thinking about that approach that is so tailored to the specific problem that your end customer has, and working backwards to solve it, how do you still take those learnings and evolve your core product and platform? I assume accumulated this knowledge and baited into the core product that you're bringing so you're not reinventing the wheel all the time. How do you make it extensible and kind of more of a platform than it is a point solution? Which, which your gross margins would suggest that you have? Exactly. I mean, that's the art of it, right? I wish there was some science to do that, but actually, you know, when you engineer, so we kind of like to think in abstractions. We like to look at the problem, think about the primitives, how does what I'm solving here apply to other problems elsewhere? And that is the art of product development in this context. But a very real example, like the work that we did to power operational warp speed, the US and actually even the UK's COVID response, that initial work was inspired by what we had done optimizing hydro carbon production for an oil major. And you can't actually, I think, fully connect the dots prospectively. Like I didn't know that the work that we're doing in energy was going to be what powered the future of supply chains broadly. But I could say that, hey, solving this really hard problem, doing it in a way that generalizes has nothing to do with energy. It's really grounded in data and how we ontologize that data. Historically, that's been an excellent bet for us, that that is going to be something, first of all, I know it generalizes from a product perspective, and it's going to be the type of problem that other people find immense value in. So this means you can go, whereas many, whereas you started with national security defense and so forth, you can go into any vertical is what you're saying, because of these generalizable approaches to solving problems. Exactly. And that's what we've done. We're now in over 50 different verticals outside of government. And in government, you know, it's not just intelligence defense, it's also healthcare, it's also tax, it's all the basic functions of a government. And within the private sector, it's everything from energy and mining to financial services and insurance to healthcare and consumer packaged goods. The industries couldn't be more varied, but if you really, it's the same platform across all these industries. And that's what we kind of honed. If you go back to the very beginning, in national security, I think the cynical way to think about Palantir is that it took something as sexy as James Bond to motivate engineers to work on a problem as boring as data integration. And that was really the first secret that we discovered, which is like, you know, we, when we approach the space, we want it to work on, I would say, sophisticated analytics that helped you find the bad guy. But that approach presuppose that you already had your data integrated. And if you ask people, they would tell you, yeah, I've spent, you know, half a billion dollars, I've integrated my data, it's here. But if you actually go to that system, you would discover it's not integrated. The real secret is that people were trying to solve data integration as a services problem. I'm going to hire an army of people, I'm going to integrate all of my data, it's going to be in one place, and then the job's done. Rather than recognizing fundamentally it was a product problem, because not only is the volume of data you have growing every single day at an exponential rate, the number of discrete data sources is growing. So every day at the margin, there's more data to integrate that you hadn't conceived of with your services-led approach. So you need a product that was capable of keeping up with the entropy of

data
in your enterprise. Having solved that problem, now you can do lots of things with it. Now you can actually go on and you've earned the right to do the sophisticated analytics. And the translation for that in the commercial world is agility, right? Like when everything was in steady state, people didn't really perceive that they had a problem. You needed something like the A350 ramp up, or some sort of event to catalyze that. And I'd say with COVID, everyone, certainly anyone that touched supply chain, experienced a simultaneous shock that got them to really wonder, like, over the last 10 years of IT investments, what have I been investing in that actually met its moment when we got shocked? And that was a kind of a sobering realization for a lot of folks. Gosh, just on the data integration, in speaking with your chief revenue officer, I believe some time ago, he definitely pointed you. So you've been doing that from the beginning, which positions you beautifully for what is happening today with generative AI. And are we right on that? That's exactly right. Now that you have your data ontologized, which itself didn't take a very long time. I would say it's a rewarding exercise to go through. The ability that you can deliver these generative AI use cases, like the speed at which you can do it, is phenomenal. And I'm thinking about a large European insurance company that built a subrogation claims agent in two days
on top of the ontologized data asset that they had in our software, or a warranty claims co-pilot that we built for a US auto manufacturer in less than two weeks on top of their data.
So the speed to value there is amazing. And so if you think about the warranty claims co-pilot, it saves four hours a day for their analysts. So half their day has now been reclaimed in a matter of two weeks. So the ROI on these things is truly incredible. How long does it take? Sorry, Frank, just to get our bearings here. How long does it take? I know it depends on the size of the organization, but how monstrous of a task is integrating all the data in, let's say, a decent sized company? Not as long as you would think. So I think one of the fallacies that's out there is really this going forward from the data. It's very, very tricky. So if you think you end up boiling the ocean, so if you're like, look, I need to get my data house in order first before I can do anything, you will find that you never get your data house in order. Or to the extent you do, you might be kind of tricking yourself into thinking if the job is done. The approach that we've seen be highly successful is working backwards from the decisions. So the kind of internal mantra we have is decisions,
not data. Let's focus on the decisions that matter this institution. Now, from there,
I can tell you what data can I bring to this decision that helps you make a better decision today and get into a learning loop tomorrow. No decision happens in isolation. So now I can look at the decision before and after that, and I can ask the same question, what data is involved with that decision? Now, this gives you a very clear priority function to go after building that ontologized data asset, where every single day, literally, you're making money doing it. So it's rewarding. You can stay motivated through it. It's not some sort of, Hey, just trust me, I got to eat these vegetables for two years, and then I promise you money falls from the ceiling, which never really seems to happen. But look, every day, I can validate that I'm on the path and the incremental work we're putting in has our way from the business. And even more importantly than that, I'm learning. I'm learning about how we as an institution get better. And so that also catalyzes the very productive and generative relationship between tech and the business, which historically

sometimes gets broken because the time horizons for delivery are very different. Business wants to know, what can you do for me today? And tech's like, look, if you only had a long term view and invested in this, give me two years, I promise I can deliver the world for you. And regardless of which one is correct, how do we solve this mismatch? We find something there's common ground and that

I can deliver for you today, and I can deliver for you tomorrow. So if we if we unpack that a little bit more kind of starting decision first, then finding the right data to plug into it, you mentioned ontology a few times. What does that really mean? And how does that approach differ from maybe, you know, maybe data warehousing or aggregating data in some other way? How is that ontological worldview shaped how you've built products at Palantir and led to, if I'm interpreting this right, a quicker time to value by using an ontology methodology, and then also how that sets up companies to action and deploy AI more effectively than other approaches. Absolutely. So we think of

the ontology as kind of the nouns and verbs of your business. It's not just the data model, which you could say roughly are the nouns, but also the actions that can happen. And so the ontology

is this operating system layer that orchestrates your enterprise. So if I want to allocate inventory, I need the noun of the inventory, the parts, etc, the model for that. But even more importantly, probably in the ERP system is my system of record for that. So I need the verbs, I need the ability to actually reach into that system and allocate the inventory. That's where it's going to happen. So now scale that out to all the transactional systems I have in the enterprise. So I have a model of the enterprise that represents how I as humans think about my business, not how that transactional systems schema happens to have been structured. My view of inventory might actually be

informed from tens of different systems. So I have some sort of canonical way of seeing it as one thing.

And then I have a single place I can go to actually permute that, to take action on it, whether I'm allocating it, reallocating it, assigning it, moving it into production, reassigning the customer orders, all the things I might want to do to it. I don't need to know I do that in this system or this system, it can't be done. Actually, I'm pulling it all together. So if you have that scaffolding, suddenly when you're doing, when you're when you're analyzing something, you're able to go beyond dashboards to applications, because I can make decisions in the same pane of glass that I was otherwise generating insights in. That's where now you can apply AI to this. Okay, well, how was I analyzing it? Now I have I have a canvas across which I can bring the latest AI and actually very cleanly tie this to an action that changes how my business runs today. And all this is happening in the framework that gives me a continuous learning right, we think of data not as exhaust, but as fuel. So all these decisions I'm making are being captured, I can see them in the context of the value chain, I can reflect on which ones were good decisions or bad decisions, I can make different decisions as my priority function changes. Maybe one day, I'm interested in maximizing revenue. Another day, I'm interested in preserving market share. Another day, I'm interested in profit, like my objective function, that's going to come as a function of the strategy from the C suite. How do I tie my strategy to my

operations that informs the decisions we actually make through that value chain? That's the value of the ontology. And I think a key part there is the ability to actually take action on it. You can't just read the data and get analytics and maybe even make predictions. That's all great. But if you can't take action on it, then you can't actually move along a process within the organization, actually get to that end result that you're looking for. And I think that's what's unique about the Foundry or generally your approach. I think that's right. One CEO I know said, I want to be better, not just smarter. And in fact, he would have rather been better and not smarter, just kind of showing you this, the latent frustration you felt on not being able to take action on really clever insights. And I think it naturally leads into how you can help companies deploy generative AI where everybody can look at the success of chat GPT and how profoundly capable it seems at first when you're asking a questions and it's responding like it's an expert and it can draft a newsletter for you, help you create a marketing campaign. But then how do you actually plug that into a process in an organization and take it beyond just this kind of like chat window? What does kind of operationalizing AI really look like?
And just to add on to that, did this shift towards large language models
interrupt your business in any way or you were anticipating it and welcome to the breakthroughs?
I'd say we always view the ontology as something very important as a key strategic differentiator. It existed in the first version of Gotham, this notion of the dynamic ontology back in 2003-2004. I think we were pleasantly surprised to see how much the world we had been building for met its moment with LLMs. It's like, wow, you actually cannot unleash the value of an LLM without these things. That's been our observation and why we've been able to move so fast. And maybe even to just take a step back to a lot of the stuff that you were saying, Frank, like I would say it interrupted us in the sense that by January, it was quite obvious that we needed to tear up all our roadmaps and get excited about how we could incorporate LLMs into our software to provide a whole new series of experiences, given where the technology had gotten to. And that we'd be positioned to do that because of ontology.
So to take a step back and kind of think about like, what is the LLM doing? Because of course, chat has captured everyone's imagination, but I might go so far as to say, I think chat is kind of an anti-pattern or at least certainly limiting in terms of the value that you can unlock from LLMs. And we could kind of say like, well, what is the LLM doing? I think it's actually really hard to say what it's doing, but I can tell you what it's not doing. It's not doing human thought and it's not doing algorithmic reasoning. It's some sort of third type of compute that is some middle ground between these things. And when we look at it, certainly through a chat interface, it's very easy for people to think it's doing human thought because it speaks in the form that we'd previously known most human thought through, natural language, but it also doesn't really know what it's saying. And on the other extreme, it's clearly not capable of algorithmic reasoning, a form of execution so precise that if the same thing didn't happen every single time, you would call that a bug. But it does seem to be instructable in ordinary pros. And so the kind of conclusion we've come through working through this is that all of the value comes from an elegant integration of human thought, algorithmic reasoning, and the LLM. And that's kind of the pattern. So when we think about AIP and what we've built here is like, how do I bring that to my customer? And to make that concrete, LLMs need tools. We all understand they can't do math. You take five LLMs, you ask it to multiply five, five digit numbers,

each one will give you a different answer. You could say that's an LLM hard problem, but they don't need to. Orbital simulation, inventory reallocation, linear programming, we have tools that do these things. You need a tool bench or a tool factory to make new tools that gives you an easy scaffolding to pass these to the LLM to use and execute.

All of the value in actually using these things, I think comes from the application layer, which I think when you look at the market, you can see the people who've jumped on this fastest owns the pixels at the application layer. And how do you bring the experiences to that? I think the reason that chat is also a limiting interface is it pushes us to expect the models to be domain experts. And if you go down that route and you'll spend all your time thinking about how do I train a new foundational model? How do I fine tune models? And I'm not saying that there's a place for that

and we can get to that. But I think what that gets you to undervalue is the parametric knowledge of the model. It's not useful for your domain. As in what does the model know? More importantly than getting the model to speak English, which of course, it does quite well. Actually, I think a lot of the value rests in getting it to speak code. And if you allow me the colloquialism, to speak clicks, to speak a JSON or a DSL that represents your application state.

So how do I take the current application state, the intent of the user, maybe with no prompt, and generate the future application state from that? They are very good at this.

Yeah. And that's, I think, the secret of driving these experiences.

Is there a world where the model gets so good at talking to these applications that you can actually, even though pixels are important and the tools that have this front end distribution, like I'm thinking of Photoshop, all of a sudden you can do really interesting and creative things in it. Is there a world where those types of UIs actually get displaced because the models get so good with a simple natural language input? I think that world could exist. I think that it's much more likely. So if you think about how our experience iterating with these things is that actually the LLM is a great way of accomplishing the first 90, 95% of the workflow. And that you're going to want the escape hatch of being able to do the last 5% with a few clicks here or there in a canvas that you understand pretty well versus having to re-attack it and try to explain what you're trying to do one more time through it. And that's the way in which I think the advantage accretes to the incumbents who own those pixels today. I do think, though, there's an opportunity for people who own the application space to integrate with other products in completely new ways. You could say at the limit, conceptually, the LLM now means that the capabilities of your software are defined by the quality of your back-end APIs and how strongly tight and type safe your data model is. The more that these things have excellent capability, the more you can actually rely on the LLM to get right. You could actually define your UI at runtime and have a fair degree of confidence that it's going to do exactly what you want based on the fact that your APIs are robust and cover the breadth of your ecosystem, and your data model allows you to validate the correctness of what's actually going on here. And that's what we've seen, that we've been bringing these experience, these generative experiences to the Defense Department in an application called GAIA that's used for mission planning. And so the ability, GAIA has maybe a decade of tools built into it. How do I do route planning? How do I do line of sight analysis? How to do a course of action generation? How do I look at the enemy's most dangerous course of action, most likely course of action? All of these

things that I'd say in some way, we've been able to do algorithmically. Now, how do I take those tools and give it to the LLM? So when the user is trying to ask a question, like, I have to plan for Dunkirk, how do I find every cargo ship that's over 150 meters, that's in this vicinity, and build an operation around that, that actually one or two sentences replaces what would have been maybe 50 clicks. One or two minutes replaces what would have been one or two hours. And in those sorts of contexts, I think that's everything. Then extending that metaphor, how do I think about how my application can interact with other backend systems instead of having to get you to go to another UI or thinking about, how do I integrate those UIs? Actually, I can be coordinating this through the LLM and the backend by manifesting those other systems as tools for the LLM.

You brought up an interesting thought or question, I guess, in my mind, which is that the commercial world, which we pay a lot of attention to is kind of going nuts for generative AI right now, and everybody's fighting to get their latest NVIDIA H100 so they can train models or deploy models, experiment in many different ways. How is the government reacting to and how government agencies are reacting to generative AI, and how quick are they moving on it?

I think there's a lot of excitement there. There's been a number of strategic investments at the top level in the Defense Department. I think there's also caution. How do we use these things for things they're actually going to be good at? I think some of that caution is very well founded, given the stakes. And I'd say in my own experience, commercial and government across the spectrum, there are good use cases and there are not so great use cases where maybe the limitations, I shouldn't even say limitations, but the nature of the technology is not understood. And maybe to unpack that for a second here, one of the things that's magic about LLMs and why I think they're kind of this third type of compute is that their statistics, not calculus. And if you're modeling it as calculus, you're going to find it, you might end up misapplying it. And so a more relatable metaphor for this might be predicting an eclipse versus predicting the weather. And when we first started predicting the weather, I think in the roughly mid-1800s, we thought it was going to be like astronomy, like predicting an eclipse. Like I can tell you when the next eclipse is going to happen 200 years from now down to the second, because it's dominated by the correctness of the calculus. The weather is stochastic. It's dominated by the error that propagates as you're solving these equations. And LLM, it's kind of got that flavor to it. It's it's actually stochastic. And so how you're going to harness that stochastic genie needs to be thought through carefully. And in particular, I think what makes it hard is that code, kind of the entire ecosystem of software development, we've grown up in a deterministic world. We're used to writing test cases, how we plan for code, how we think about the what it's going to do deterministically. And now when you introduce even one stochastic variable into what was otherwise a deterministic system, the whole thing is stochastic now. So there's a big part of the tool chain that engineers need to rethink and kind of even more than that, I think probably an intuition of where the value lies and how to use these things that engineers need to rethink as they go through it. And that I think is where the caution is well, well warranted. Now on the other hand, my consistent experience with this has been, it is truly empirical. You cannot think your way

through where the value is going to be or how to use these LLMs. You have to get in there, roll up your sleeves and attack the use cases. And I think on the other hand, that's very exciting, certainly as a practitioner. But it kind of speaks to, you know, go in eyes wide open, understand and be conscious about it, understanding you're dealing with something that is statistical
and not calculus, but then get busy at going after the value.
Can I ask you, we're trying to figure out many, many people in our business are scared that we've entered another tecton telecom bubble, a repeat. That's not our belief, but I do think when there's such a rush of capital into a new idea or into a breakthrough that there are inefficiencies and mistakes. You're suggesting that this is true. So what is your, and I'm sure you've had experience with your own clients, are they still pushing full throttle or are they saying, wait a minute,
we need to think about this a little differently. We may not understand what this is much in the way you've just described. What's going on out there? I'd say for every institution,
you have a little bit of both going on. Probably it leans towards momentum. I should say maybe a lot of momentum and a little bit of caution. And we've been really thinking about how do we help our customers understand how to think about these things differently. Like where are the risks? Where is it quite safe? One of the most effective techniques for us to the point of this journey being truly empirical is that we've been organizing these boot camps. We go city by city. We've done one in Chicago, one in Tokyo. We get 10 to 20 customers together. They spend a week building a production application. So they are going to leave that week having shown their peers what they built and having shown themselves that in a week they can build a production application using this technology. The most important part of that journey is not only the power of belief and what they can do, but the intuition over, oh, this is how it's actually going to work. Because it's not going to be a chat interface, realistically. That's not where the useful production outcomes are going to come from. What does it really look like? What sort of use cases should I go after? And I think one of the common mistakes is when you think, to my earlier dichotomy
of algorithmic reasoning or human thought, if you're trying to get the LLM to do either of those two things, it's a safe bet that your use case is going to fail. But if you decompt your problem in a different way, you can get a lot of the value. So to the warranty claims example that I was giving you for an auto manufacturer, what the LLM was doing that was quite magical is reading an unstructured claim coming in from a dealership filing the warranty claim and figuring out which component in the car the claim related to. So that process alone for the tens of thousands claims that come in every day took four hours a day per person. But what the value for the auto dealer is taking that data and analyzing and looking at what parts do I need to manage? Where do I need to do recalls? How can I get ahead of this? How can I reduce my cost of quality and cost of warranty and the accruals that come from that? How do I manage my suppliers based on the company? So how do I take the strategic decisions? How do I leverage the human thought that is so nuanced and so contextual and apply it towards that problem? And how do I use the LLM to do something that otherwise was actually just getting in the way of unlocking that value?
So for all these use cases I look at, I think there's quite a bit of art in decomping
that elegant integration between human LLM and algorithms.
I don't, I'm just picking up on the word art. I might be wrong on this, but I don't associate

engineers and art. I could be completely wrong here. Maybe I just don't understand. Are they comfortable in this zone? Intuition, art, you're using words that I don't normally associate with engineers, but I could be wrong. Yeah, I think we could go to a number of directions with that. Well, I think first of all, the methodology for deployed engineer means that you're sitting right besides the claim analyst and you really get a sense of what are they doing? What are they not doing? Where is the outsized value coming from their human judgment versus where is the software getting in the way of them actually creating value? So that's a piece of it. You're exposed to the customer's craft, if you will, that allows you to model it. But I think even more profoundly, to speak about culture for a second, I think culture of a company is hard to say. In some sense, you have to experience it. But I'd say that the single most important tenet of Palantir that our culture is a symptom of is this idea that Palantir is an artist colony and that comes straight from Alex Karp himself. For better or worse, most companies are factories. Hire you for this sort of role, you're this sort of cog. If you do this role long enough, here's a ladder you can climb. It's very well defined. You can even see it in our projection of normalized titles at the company. We have all these weird titles for what you are because we don't want you to import preconceived external notions of what your job is. Really, your job is based on who you are. We are going to go on a journey together to define what you're good at and not good at, and build a role that's end of one. It's built for you. I think a part of that is you don't get dali to be better by saying paint more like Monet. The art is unique to the individual. I think this is a place that values craft immensely and tries to tease out in our individuals how they can see, of course, the science and rigorous engineering of what they're doing. But understanding it that it's this sort of dialectic with the art that makes it truly valuable.
Sounds a little like investing. Yes.
Do you, like thinking about inside Palantir and pulling the onion back, do you use these tools and an ontology generative AI inside the company? Absolutely. I mean, we've always dog food our products and we use them religiously. I think one of the exciting things about generative AI is how quickly we've been able to build our own co-pilots and to go after problems. How do I auto generate documentation from the things that we're doing? How do I build co-pilots that help us do incident response better? How do I transform the internal operations, the kind of GNA stuff that we're doing? That gives us ideas of new things that we could even be going after. You think about all the video conferences that happen and the audio that's there. So one of the first things that we notice working with companies, companies like Panasonic Energy North America, they make every battery in the Teslas co-located with the Gigafactory was that you have tens of thousands of pages of documentation. How do I maintain equipment? How do I do X, Y, or Z? You go to the Defense Department.
You see the same thing. Like they have O plans. So like how do we expect to fight this conflict? Thousands of pages or with drug discovery or big pharma, they have statistical analysis plans that they go back and forth with the FDA on about a thousand page PDF. So there's this rich human language that you can apply LLM's to to do so much. One of the constraints is understanding where is that document actually stale? Who likes writing thousand page PDF documents? Who likes reading those things? So once you get into the nitty gritty and to the point of secrets that are at the front lines, you start realizing like this document was accurate two years ago. It doesn't really reflect how the workers believe things are happening today. What does reflect that

is every single Slack conversation, every single JIRA thread, every single if you could just somehow record the audio of the incident response meetings that are happening. That's where the true knowledge
is. Now that's a feasible problem because you can convert the audio to text. You can start driving that through LLMs. You can auto generate question and answer pairs. You can find you in a model on
that if you want. You can start bringing that knowledge to bear in the next operational context and setting. And so I think it's expanded in some sense. It's allowed us to kind of feed the iterative feedback loop of data integration. Like the core thing that we believe creates value. It's like, oh, we didn't even realize there are all these data sources out there that were so unstructured before. It wasn't feasible to really integrate them before. They now are. The efficient frontier has been pushed out. I can bring so much more of what was historically viewed as ephemeral knowledge to
pair against the problems that we face. And that's what we've been focused on internally as well. You know, it's interesting as we, in the investment world, we try and figure out, okay, where, what's going to be commoditized? And, you know, you try to avoid that part of the market. And where is the real value going to be created? And, you know, of course, everyone is checking the box with NVIDIA, right? GPUs and so forth. But the ramifications of what we're talking about are so profound and pervasive. So do you have any thoughts on commoditization? We were thinking about foundation models, commoditized or not. And, you know, are the real winners those with truly proprietary data? And the example you just gave, yes, you know, many people have wondered why we own Zoom. You know, it has incredible data. It's just announced a bunch of AI products and it's going to have Zoom Topia. So we're just trying to figure out, okay, everyone's a winner from a productivity cost control point of view. But there is going to be some commoditization. How do you, how do you? Yeah, it's an excellent question. I think probably
the value accretes at the bottom and the top of the stack. That's my current intuition. So, you know, you're going to need lots of compute to do these things. And then I think the magical experiences are going to come in the application layer. And so I think that the middle, the models are going to be squeezed. I think they're probably going to be a few really big winners there. But there's probably not going to be many, many, many winners in that space. And a big part of that is like if you don't own the application layer, it's very hard to create the experience. And you see that if you interact with the model companies and you start asking questions like, where's the value, you know, what sort of use cases should I go after? It's really, here's our API. There's not enough opinionation there. And it's not there, you know, how could they have earned the opinion yet? Maybe they'll get there and I don't want to pollute that. But you kind of, I think the ones that are being very successful are moving up into the app layer. They realize that they need to own that app layer to really provide the value and be able to capture enough of that value. And I think it also explains maybe if you just wanted to look at this as an outside observer, the speed at which Amazon, the different approach that Amazon has taken to this than Microsoft. When you really look at like, where's Microsoft super opinionated and where does the opinion seem magical? It's how are they going to bring these experiences to office and to the applications they own? And I think Amazon looks at it and says, look, let's just have lots of models running compute because that's how they make money. And that there's no opinion because they don't have the app

layer to have an opinion against. That's really interesting too. Who are the sleepers here, do you think? Companies we're probably not thinking about. I don't know. I think I need a little bit. It's so, the space is so fast evolving. And of course, the amount of funding that's coming here is, but I kind of suspect that the value accretes to the incumbents in AI. That you're going to, this is a, you know, and to the extent it doesn't, I think the incumbents were asleep at the wheel. Like the easiest place to go capture incremental value and deliver value for your customers is in the pixels you already have shipped to those pixels today, go after it. There's nothing about AI that creates a disruption and distribution. And that's why I think it's going to be very hard for complete new entrants to have a big impact here. They're going to have to find some way of integrating it to the value chain. Interesting. Because the productivity benefits should accrue to many, many companies, right? Certainly. If you just think about it from the vendor side of this. So I think the folks who have the first crack at this are going to be the ones that own the application layer. We've obviously seen the uptick work from NVIDIA, because that's going to lead, right? Like, okay, I need the compute to even get to the starting line. Sure. Then once you're at the starting line, what's going to happen? And I was with a very senior person in government the other day, and he was just telling me, you know, the more I use chatGPT, the duller it gets. That's not really a knock on chatGPT. What he's experiencing is what I think lots of folks are experiencing are the limits of the parametric knowledge of the model. Yeah. Okay. It doesn't know anything after 2021. And, you know, it's okay. So what's the utility I'm really going to get for it? Well, I think what he's really needing, what he's saying is I need this inside of my applications, the things where I go to do work today. So who's going to be the quickest to do that? So if you're a total disruptor, you got to go build that application compete head to head on an existing application and then bring the experience. That absolutely might be possible, certainly for certain wedges of the market that you can grow into. I just think that the first mover advantage is going to rest with the folks who own those pixels today. Is there anything about existing technology infrastructure? You're mentioning the incumbents, which would limit the efficiency or the effectiveness. There is controversy out there on Microsoft. Is it still so legacy bound that there's going to be something to trip it up for that reason? I've been moving pretty well from what I can see, but I'm just throwing that out as an example. Yeah, if there are, I don't see them yet. I wouldn't bet against them as an example with Microsoft, certainly. And I think that in general, the infrastructure is probably not going to be rate living. Of course, everyone's having a hard time getting at GPUs, but I also think the amount of movement and open source, particularly for Meta with Lama 2, is creating a whole surface area around which the models are not going to be the constraint. I guess another way to frame or just react to Kathy's question, there's the infrastructure for deploying and training and doing whatever you want to do with the AI, but there's also the ability to connect it to the rest of your organization and really integrate it rather than just make a standalone kind of new AI product. It's easy to build modern cloud microservices architecture from scratch. But when you're integrating it into a legacy system that's running on a mainframe somewhere that hasn't been updated in 20 years, it gets a lot more difficult. And there's probably some organizations that are better positioned to move fast. And the incumbents being the large tech companies

probably can, but there are other incumbents in other industries that probably are not as well positioned. That's a great point. And I think it goes back to what I was saying earlier about how robust are your back-end APIs and how type safe is your data model?

Type safe is your friend with LLMs. To give you some intuition for that, you could say in some sense English is the least type safe language. The number of sentences I can make that are grammatically correct for garbage is probably larger than the space of correct sentences. There's a lot of the risk of hallucination becomes very high. The more type safe your data model is, the more you can safely rely on the LLM filling in the parts of the data model that you need. And it gives you a shortcut to leveraging them and accelerating their deployment of production. Sorry, can you type safe? This is not a phrase that I have used.

Yeah, sorry. It's a computer science term for you could say, I just have a string of words. Here are words. It's very hard to validate, are those words correct? Do they form some sort of structure? You can say that's like the least structured way that I could answer a question for you. On the other extreme, you could have something that's highly ontologized or here's a precise data model with a hierarchy and the answer must be provided in this very specific form. It's how I represent the world. Now I can validate that this is a possible, this is a legal or illegal state that can exist. So now when I'm leveraging an LLM and I'm trying to get it to generate the data, I'm not just saying speak a sentence, which actually maybe only a human can validate as being correct or incorrect. I'm saying produce this data structure, which now the computers can validate on their own as being right on or garbage or needing to re-attack it and ask the question in a different way, etc. And then you can start reliably building, you can now it's composable into building blocks that drive your software. And so I think that's where it becomes a prerequisite. So if you've taken shortcuts and how you model your data, too many things are done by convention, as opposed to being done explicitly, you're going to find that you're going to first have to fix those things in your data model before you can truly get the value from generative AI. And that's Kathy, what really speaks to your point of, okay, maybe they're legacy providers, maybe their software that can't keep up. I would focus on these two dimensions as being the greatest risk, which is, you know, how much of this is kind of sloppily organized between front end and back end APIs. It's not clean enough that the LLM can grab it and get leveraged from it. And the other part is how modern is the data model and how specific is it? And I think like the emergence of generative AI and the excitement about it is probably, and the requirement to have this well documented and enforced data model is probably driving a lot of interest and business towards you with AIP as being an enabler for that, that it almost, and this goes back kind of way in our conversation, but you've been building towards this type of infrastructure for an organization for a long time. And now generative AI seems like a really good fit for it. It's a really good point. We, you know, people used to ask us all the time, like, why do I need an ontology again? And that's always a dangerous place to be in. It's like, you're saying like, look, I'm telling you, this is, this is, you have this orchestration layer, if you're a reason, that question has gone away. It's just very obvious when you're trying to build an LLM backed function, okay, well, I need a way of orchestrating tools handling into it. How do I, how do I do that? I need a way of not hallucinating data, but actually using traditional information retrieval techniques to answer this question. And so the value of ontology as a prerequisite

to deploy this in production, it's just, I'm obvious to folks.

So you are the, have just become chair of the board of Ginko, a company we're really interested in, but is not well understood out there. So many companies in the innovation space are not well understood as they're evolving. And in fact, many people, you know, it's, it's kind of a, I don't know what to do with this. It's all kinds of industries and, and, and data, and you've got the healthcare world, and, you know, they're very precise in terms of what they want to analyze. And this is not it. So just, just wanted to know how, how you and Ginko became connected and how you would describe it to someone. You, you, you describe things very, very well. We've learned in this podcast how you would describe what Ginko is doing. Yeah. So I, while I just became the chairman of Ginko, I've been on the board for eight years now. And so I've known Jason and Reshma and Barry and Tom and Austin for quite a while. So I probably started getting involved with them when they were about 50 people. And so I've seen the business go from a place where they were really in a zone to help industrial bio manufacturing, like how we build flavors and fragrances to a place where now they're helping actually manufacture drugs. And it's been quite exciting. But what's been consistent the whole time is that the mission of Ginko is to make biology easier to engineer. And if you kind of squint at it, you can think of a cell as a computer, you know, you put in code and you get out product. And we don't, it's actually, you know, part evolving understanding of how this computer works is growing. But that's, that's the thing that they're going after. How do I go from a place where this is just an accident? We understand what we understand

because it happened naturally to a place where we can control these things. And as a consequence of

that, I think in many ways it is the future of manufacturing. You know, we have a kind of very traditional view of thinking about how do we make things. But here you have a place where you can program a cell provided feedstock, like sugar water, and you can get product out of it. And many of these things actually speak to the core of the human condition. How do we manage the environment that we live in? How do we manage the recovery of rare earth minerals? How do we think about health? How do we actually manufacture? This is what we experience with COVID, where it's

one thing to be able to discover a vaccine or a treatment. It's a whole another thing to think about how do you produce that at scale and on a timeline that actually matters. And those processes are biological, actually. And so there's a tremendous amount of potential that's coming out of this. And I think one of the greatest opportunities for Ginkgo right now is around generative AI. How do you teach a large language model not to speak English, but to speak ACGs and tees? And there's a lot of reason. I mean, like you can think in many ways nature is not that good at it. You know, you could say that English is something that we as humans designed. And yet the LLMs are still really good. On the other hand, we didn't design DNA or ACGs and tees. We're barely understanding it. And our ability to manipulate it is very bad. So what sort of alpha or upside can you expect from an ability for a multibillion parameter model to actually understand what's behind this? I think a lot, actually. And then Ginkgo is uniquely situated then because they have this integration of let's call it the AI, the dry and silico side of this. With the wet lab, the foundry as a highly automated series of operations where you can come up with

a piece of DNA, robotically put it into a cell, produce the output of that, test what's come out of that, get feedback on what could work, create a new design and do all that in a closed loop environment that's subject to a tremendous amount of automation. It's almost just unfair how fast you can go. And I think folks are really seeing that in terms of how it can transform not just their cost of R&D, but how many shots on gold you get for the money that you're having and how much can you advance fundamental knowledge. And on top of that, they've built a tremendous code base.

So you're not starting from scratch. You're not starting with the knowledge of a single institution. You're actually standing on the shoulders of a collective understanding that Ginkgo's developed over the last 15 years of DNA that's growing so fast. So their actual internal code base is, I think, roughly 10 times larger than the aggregate external publicly available sources like the uniprotts of the world that people have used to generate alpha fold. So you think about what that means and how fast it's growing. The kind of compounding capabilities are quite exciting.

Yeah. That was a great explanation. And an awesome convergence story between AI and life sciences that I think is exciting. For me personally, I've always felt like there was some kismet there because the first meeting I had with Ginkgo, they told me their product was Foundry.

I was like, how could that be? How could that be? This is amazing.

Yes. No, it's an unbelievable story, company. And I think the financial markets are having trouble really getting their arms around it. And the proof will be in the pudding.

I shouldn't have even used that expression. That is so old fashioned, given what we're talking about right now. But anyway, thank you so much, Shom. This has been fabulous. And I am really thrilled that we're able to bring you on this podcast so that you can really explain actually two stories that are not well understood in the marketplace. And they're both so exciting. And I guess what I'd like to end on is Alex says, we will be the largest AI software, I don't know the exact words, but the largest or the most valuable, maybe that's what he says, AI software company in the world. Clearly, you would have said that an incumbent like Microsoft is in a very good position. So maybe just if you could end on that, when he says that he says it with such conviction. And part of it is we've heard that Palantir understands the way technology is going to be in five to 10 years. Is it something in that next five to 10 years that that is going to be the aha moment for people who just do not understand this story? Well, in many ways, I think we think of ourselves as incumbents. So I should kind of clarify that in a sense where what we're talking about is do you own the pixels that make the decisions and operations today? And or do you have the ability and the distribution to own those pixels and deliver on them? So the fact that within a week, we can be working with net new customers to get production level use cases and capture their imagination. I think what we feel the reason we are so excited is we feel like we have the elegant integration between the human thought, the LLM and the automatic reasoning that's required to actually capture the market. And most folks are going to be stuck in one of these modes for the other, depending on where their platform has limitations. And so in that context, we can go incredibly fast. Many of the aspects of generative AI have meant the parts of our process that were bottleneck are no longer bottleneck that we can reimagine aspects of how we go to market instead of doing a what would have been otherwise historically really fast, the six week pilot, we can now do a two day pilot.

You know, you think about the opportunities that open up and how much more quickly our customers
are able to take this technology and build their own use cases for themselves. So I think there's
a lot of this probably like five or six components that feed into the intuition that Alex has,
which I share that really we can take the whole market. Love that. Yeah, well, this was great.
Thank you so much. And thank you, Frank, for teeing up those great questions.
Thank you guys for having me. Okay, thanks.