Software in the Age of Agents | The a16z Show

60 MIN · YOUTUBE · HTTPS://WWW.YOUTUBE.COM/WATCH?V=MXS4ERDXOEE
https://www.youtube.com/watch?v=Mxs4erDxOEE

SUMMARY

The discussion on the A16Z podcast centers around the evolution of enterprise software, particularly the concept of "headless" software and the implications of AI and agentic systems on traditional workflows. The speakers explore how software's stickiness has historically been tied to human interaction and the complexities of enterprise processes, emphasizing that as automation and AI become more prevalent, new opportunities for startups emerge to bridge gaps and enhance efficiency.

- *Headless software shifts focus from user interfaces to the underlying data and logic, acknowledging the changing landscape of software interaction.*
- *The misconception that simple databases and APIs can replace complex systems like SAP overlooks the intricate business logic embedded in these legacy systems.*
- *Stickiness in software is driven by user dependency, compliance, and the ingrained processes within organizations, making it difficult to replace.*
- *The rise of AI and agentic systems presents opportunities for startups to create solutions that enhance existing software rather than compete directly with established players.*
- *Network effects in enterprise software are challenging due to compliance issues, but internal network effects can emerge as employees find innovative ways to leverage new tools, enhancing collaboration across functions.*
- *The integration of AI can facilitate communication between previously siloed departments, creating new categories of software that address organizational inefficiencies.*
- *Startups should focus on capturing the "data exhaust" from existing processes to inform new solutions and improve operational efficiency.*

There are many things that made software sticky, but a lot of it had to do with the fact that the way a human interacts. In an agentic world, do you actually need that? The data, the logic, everything stored below it is really where the value is.
>> There's this wild underestimation about like you could vibe code your way into enterprise software. Larry Ellison at Oracle, he went on a rant about how enterprise software was so stupid because everybody customized it. The minute you automate the most mundane

thing and think you have it all squared away, whole new things appear.
>> Misconception right now is that you can just have you know Postgress database and APIs and then bam like you can replace SAP. That's like absolutely not true. That piece around the logic and everything else that is encaptured in SAP is way way more important than the fact that like oh this data just happens to be in this database. One of the things that happens in technology shifts is nobody understands exponential when it's happening. The biggest opportunity right now is
>> Welcome to the A16Z podcast. I'm here with Sema Amble, a partner here on the enterprise team, and Steven Sonowski, who is a board partner at E16Z, as well as a former uh member of of Microsoft, friend of the firm. Um, and here we are today to talk about um a piece that Sema wrote about a month ago called Is Software Losing Its Head? Uh, and this piece I'll I'll let Sema talk about it in her own words. Um, but this piece was written a couple months ago. Salesforce announced that they would be going headless. Um, and today we're here to kind of discuss, you know, what does that mean? What does that mean for, you know, the future of SAS products, the future of, um, you know, just just kind of software more generally? So, Sema, um, can you just walk me through first what headless software means and explain kind of what changes it introduces?
>> Yeah. Um, so headless software has really it's it's not a new term, but I think has really risen in uh in like the you know public domain of of interest in in a topic that people are talking

about. One of the interesting news points has been Salesforce making this announcement. they were launching headless 360 which was really in classic Salesforce motion you know uh history a uh a marketing announcement more than anything else but it does capture um it sort of it's an acknowledgement of what's happening which is you know traditional software had been built around humans accessing it and um it was workflow to capture data and we could talk more about what that meant in an agentic world do you do you actually need that the UI doesn't matter matter because the agent isn't accessing the software via the UI. We could unpack whether the UI matters or not. But in the idea of the being headless is the the data, the logic, everything stored below it is really where the value is, not just the workflow software that's being tracked at the top.
>> Gotcha. Um, and uh, this was announced a couple months ago. I'm curious kind of in in the past couple months, what have we seen? I mean you wrote even in the piece kind of you at the beginning you were a little bit you know funny maybe and you said is this really even that big of an announcement? Is this sort of a rebrand of you know APIs that they had kind of already made available? So does this feel like a significant change? Is this maybe more of a branding exercise? Um and kind of like what have we even seen in the past couple months since we've been able to kind of observe what changes have have really happened? So I'll separate it out into the Salesforce context and then like the broader context which I think the broader

context is more interesting. In the Salesforce context probably not that interesting. Uh I think Salesforce I think rightfully again it was acknowledging a shift that's happening in the market. I don't from what I could tell nothing actually changed. Their 360 product was uh the same APIs that had always been exposed now rebranded as as as their 360 product. How and and APIs have always existed. So you know many uh or I shouldn't say always but for a long time have existed. Um but I think the broader trend here is there you know Salesforce among others are thinking about how they build themselves for the agentic world. Um and so if an agent needs to access the data in a CRM like Salesforce um what are they doing it? Are they doing it via the UI or are they using the API? And that's the you know Salesforce is saying like okay hey we know what is changing that there are agents who are needing to access the data. Let's look at the you know let's offer a headless version for them to interact with um the data versus going via the UI. That said again I don't think anything actually changed in the Salesforce context but Salesforce is in the old one. Another example is notion has a headless product and actually I think that makes uh even more sense because it's much hard like I think many users of Salesforce are probably less technically adept less likely to be building their own agents although there are many many more people who are doing that with Salesforce notion users tend you know all things being equal are more I'd say techsavvy and more agentic as as builders and I think notion is is one of

many other companies that is also trying to figure out okay what is it that I offer for um and you know how do I make you know what APIs do I expose? I think Stephen will talk more about MCP. Um and and again I think a lot of this is also getting caught up a little bit in nomenclature and like okay what are we calling things?
>> Um and I I I think that's one thing but I think the the broader trend around how agents um access systems of record I think is the the bigger point.
>> Yeah. Yeah. And from my understanding it's also I mean it could also apply to something as simple as a chatbot. you know, it's not necessarily just an API or an MCP server. You know, you you could be, you know, Salesforce um acquired Slack a couple of years ago and like it could be something as simple as like you you sort of interacting with with a CRM via chatbot.
>> Totally. And I think I read somewhere that there's been like a 300% increase in Slackbot uh Slack agent usage. Yeah. which is essentially saying that you don't need to log into the Salesforce EOS uh interface to get the data or whatever data it is. So yeah, it's it's essentially again all these are agentic ways of accessing it versus the human needing to go in log the data or or um from a read perspective go back and see okay here were the like here's this opportunity here's what happened and um look at them themselves um and so the that interface is is less relevant.
>> Yeah. And Stephen, do you have anything to add here just on the kind of like definitional territory that we're covering right now or or kind of this

discussion?
>> Well, sir, I mean like we're in definitional hell right now where is is,
>> you know, part of a new wave of technology is you make up a lot of new words for things that you kind of did before. And that's just a natural part of technology evolution. Um, but I I actually think it's super important. Like first you have this, you have a agent, which as far as I can tell right now is also a new word for program that takes a very long time to run and might not finish. And and so yet that's just like a a branding that's the best branding ever is to call a program that's takes a really long time, which we used to call a bug, is now like the coolest new feature ever and it's just now an agent. But in in seriousness, the most interesting way I think to think up what you're really talking about doing differently between an agent and an API is really what are you actually doing? What is the agent itself doing? Is it looking something up? Because that's actually a pretty lightweight thing that all systems are pretty good at. And in fact, many many of the newlyannounced, you know, headless agent APIs are are just lookup and they're just you basically have a new interface to the old way to look something up, which is a lot more forgiving, a lot less UI goo and stuff like that. Then there's like I want to do something and that's where you get into very interesting issues over like well if you do something you have to be impersonating a specific person you have to have their credentials like it's a very is it another paid seat is it the same paid

seat you have all these interesting enterprise software issues that come up if you actually want to cause a change to a system of record and then there's the third thing which is analyze. And so analyze is more than look something up. It's actually look up a bunch of stuff. It often involves multiple systems. And that seems very very tuned to an agent because you you're not time bounded. You can spend energy iterate. You can route it to different models and get different answers back and compare them. But it's also where hallucination really is a huge issue because if you're going to go and analyze something, you actually need a way to verify that everything every step of that analysis was correct. And so I think it's super interesting and important when you look at headless and agent which are conflated in you sort of have to figure out what you're talking about because we're on different places in the evolution the learning curve and the deployment of agents relative to sort of that three-way matrix
>> matrix.
>> Yeah. I I think this is actually a good leadup into a follow-up question, which is kind of like historically, what has made software sticky and how are agents starting to disrupt that? And I I leave that to either of you to answer. Maybe you guys can both kind of debate about that. Yeah, I'd say there there are many things that made software sticky, but a lot of it had to do with the fact that um it was built around like the the way a human interacts, right? So um the UI was sticky because you know the number of times you had to read and write the frequency of access the downstream

workflows all of the like undocumented like you know what we call SOPs or uh standard operating procedures all the stuff that happened around the software that got ingrained in muscle memory and process and then external parties etc. So like a CRM may be sticky because a sales rep needs to go in and out of it all the time. They're used to interacting with Salesforce. A lot of times when like new VPs of sales come into our companies, they they're like they mandate that Salesforce is there because they're used to using it. Their teams are used to using it. Um and then there's, you know, finance may rely on the Salesforce um output for billing and upstream marketing is going to rely on it as well. And so there's these dependencies. Um and these all are driving stickiness. But I think the other piece too is there you need one single set of truth, right? you need to know like an account is closed and who is working on it and all of that needs to be logged in one place. Um and I think if you go from CRM to say an ERP or payroll like that absolutely has even like legal reasons and compliance reasons why you can't have um numbers that are not being you know tracked uh as cleanly and and correctly as you know an auditor might like for example. Um so anyways this all drove stickiness there and and and durability because you know you were used to using Salesforce the whole ecosystem was using Salesforce and it was the default option and maybe you know there's one or two others in the market. Um and so I think historically those were some of the things that were were driving stickiness.

Yeah. I mean I those are all exactly right. I I think it's important to also consider that the most sticky thing you could do is actually collect money from a customer. And if you're collecting money, it turns out it's really really hard for them to stop sending you money. And it's really hard for them to figure out what to do if they stop sending you money. And and it it sounds really trit, but the the stickiest software is software that's getting used somewhere. Then when you dig in and try to come up with reasons, well, it just depends on who you talk to in a company, you know, you talk to the HIPPA compliance people in some company and they're going to tell you this is the software you have to use because it's like the most bestest HIPPA compliant software. If you talk to the administrators, you're going to hear about onboarding new users. If you talk to the users, you're going to hear about muscle memory and keystrokes or labor unions or whatever. and and so you have to be yet you you really want to get the software sold and that's your fastest path to sticky and after that it it's sort of you know a winner's tail over over what caused it to be sticky and and in fact the best thing about sticky is if you're the rep for a company that you've sold something to and the company is threatening you like hey we're going to replace you we're just going to listen to them and you're going to find what's sticky. And if that works like three or four times across different accounts, then you've just told the tale as to what made the product sticky. It doesn't matter what the PMs or what anybody else thought of.

It could be some crazy arcane thing, you know, like I have stories of lots of sticky software and lots of arcane things, but like anyone who's ever tried to displace um Microsoft Outlook as email very quickly learned about delegate access and and having calendars owned by multiple people and all of this crazy stuff. And like I can tell you there was no meeting where we said, "Okay, let's figure out how to make the calendar the sticky part of Outlook and make sure we handle recurring meeting exception handling well." And then you go and you find out, you know, General Motors isn't going to displace like 600,000 seats because of the calendar or some crazy thing like that. And so you can, it's really amazing in enterprise software what causes sticky and how you can actually capitalize it on it when somebody threatens to take you out of the out of the enterprise.
>> I I think there's a really good point there in two things. Inertia is a really powerful force.
>> And then I think the other thing is yeah, nobody when they're building the software, they're not like thinking about like this rubric we put together and like tick tick tick. We got all these features that are going to do all these things. But I think the practical reality is also as software extends its tentacles across an organization and it gets ingrained in people using it and they've been paying for it for a long time, it just seeps into how people are doing things and that's like hard to rip out.
>> Yeah. Yeah. You even talk about this in your PC mode where there's sort of all

of these like invisible tacet sort of understandings about how to use different products or things that are embedded both within the software but also within the people using them where it's just like it does become hard after a while to uh to extricate yourself from from whatever ecosystem you happen to be in. And in fact, Stephen, I think you you've even you know said the SAS apocalypse is overblown. you've written an essay called the death of software. Nah. Um where you where you you know emphatically sort of rejected this idea. So maybe if you want to if you want to maybe just recount that piece a little bit for us.
>> Well, I mean Sema co-wrote a post on on SAP which is sort of the ultimate ultimate example of sticky software. I mean there is nothing the the only software that's stickier than SAP is behind the scenes and it's the software that insurance companies wrote and they wrote all this software like 50 years ago or 75 years ago and if you ever there's no replacing it like it it just it in fact you whenever jokes come up about like businesses that are looking for cobalt programmers it's to go and work on the insurance software that exists in in every state of the union and and in many ways what you're seeing like with the one of the biggest successes to date in Stripe has been somebody actually went in and for the first time in it two generations coded up the software to collect money from people which itself had previously been an unsolved problem on the scale of insurance because nobody put together the tax laws for every country every

every jurisdiction
every locality, every border crossing,
every currency exchange like it it it's
mindblowing that. And so now that is the
most sticky. So like that is not going
anywhere ever. Like it it'll be like
we'll be doing this podcast with like
great grandchildren 100 years from now
talking about how sticky that experience
was. Just like I told like oh you didn't
know this but the software that runs all
state is older than me and it's not
going anywhere and that's because the
these examples are ones that codified an
external force and that external force
was the regulatory body that they
embraced the seamless examples of SAP
like it just codified a company and and
so like if you take SAP out of a large
automobile manufacturer
there's no automobile manufacturer left
or or Walmart like the company just
evaporates
because the the company is defined not
just by purchasing the software not even
by just using it but by how that they
codified the business rules into that
product. I think it's a good point to
double click on because I think a
misconception
right now is that SAP okay well you can
just have you know Postgress database
and APIs and then bam like you can
replace API replace SAP and that's like
absolutely not true. I think partly I
mean Stephen I don't know if you want to
elaborate on or not I'm happy to but I
think it's that piece around the logic
and everything else that is h like that
is encaptured in SAP is way way more
important than the fact that like oh
this data just happens to be in this

database
>> there's a reason why also SAP like takes you know multiple years to implement and get it's not because like oh you know yes the system integrators are slow and part of it but it's it's customized to the way that business actually operates. Um, and I think that's like an important part about why you can't just obscure away the software completely from and and turn it into a data database plus APIs. Yeah,
>> this it's just so important because this is one of the things where like startups look at enterprise software and they think about it in terms of startup scale and so they take something like mundane like expense reporting and like okay well we have 40 people and like you know one person could figure out expense reports for 40 people like you you could hire a human and that and be done with it. that like literally you come back from a trip, you dump the receipts in a bucket and one human rifles through them and the expense reporting problem goes away. Or you say, "Oh, forget the human. We'll just all take pictures of our receipts and OCR it and categorize it and the whole thing will go away." And and that's fine until you have a h 100,000 people in 20 countries with different national laws and policies about business expense and all of those and then you overlay corporate policies and and the whole thing just and then your business is just codified that way and it and you can't replace it. You back in the um in the late 1990s who now is sort of the godfather of enterprise software Larry Ellison at Oracle. back then they were only a database company

and entering the world of Netswuite and ERP and all this but he went on a rant he a multi-year rant about how enterprise software was so stupid because everybody customized it and he had this saying that just said businesses should just stick with the 80% solution and they should just use whatever works like 80% of the time and most enterprise people were like well a you're just talking your book because your software only does 80% of what I need. But B, like that's that's just not how it works. Like if you take the auto industry and you just take the top 10 companies in autos,
>> they they all, you know, putting aside EV versus gas or whatever, they all just make cars, which is a lot of known technology with assembly lines and workers. Like what differentiates the the companies and what differentiates them is how they operate and the internal processes to decide what car to make, how many more materials to buy, what currencies to hedge, how many people to hire, when to introduce a new product line. All of that is enterprise resource management and planning. And how is all of that done? It's all in SAP. So those companies are effectively run by people sitting in conference rooms looking at SAP screens and and the difference between Ford and Toyota and General Motors and Daimler are just are not just that they're looking at the same screen. it's that they chose which screens to look at, which customizations to make in those screens, and then they go and they buy steel and aluminum and wire and dashboards and radios from all the same

places. And so it's a I I just think
that people wildly underestimate
the the level of house of sophistication
that customers apply to this software.
Like when back when we were first
starting to get Excel used in companies,
you'll laugh at this even like like the
we used to do these little visits and we
go visit bankers and so we're sitting in
Goldman Sachs and we're telling him
about Excel is better than Lotus 123
blah blah that's super old. You don't
even know what 123 I know. Trust me,
it's old. And and the guy at Goldman
looked at us and said, "I don't think
you understand. We make more money from
Excel than you do."
and and you're and we're just sitting
there like what is he talking about?
Like it made no sense to us. And then we
started to think about it and it's like
well we sell Excel to Morgan Stanley and
JP and Chase and everybody else and what
Goldman was saying is their application
of Excel is so differentiated.
A and that's and that wasn't just people
typing. They built add-ins. They wrote
all this code. They defined their
WordPress. know there's this wild
underestimation about like you could
vibe code your way into enterprise
software. I uh was at a dinner last
night and there was someone there who um
was like the head of revops at a I don't
know maybe growth stage startup and his
task this is like a thousand plus person
company was to rebuild their Salesforce
instance internally
and you know I think he's like oh well
you know we know all the fields we can
import all the data and I was like
that's not really the part that's tough

right it's well how are you deciding what like what gets captured how the whole organizational lines around it. Uh, and then who's going to maintain this also over time. I think that's like a piece that just gets falls off. You know, you can you can vibe code a CRM. We've all vibecoded projects that have like already gone stale and know we haven't touched again because it's painful and it's it's it takes time and needs to adapt the business. Mhm. And and Simo, you've also written about how, you know, there are there's an entire ecosystem of startups now that are just building on top of SAP and that are, you know, like sort of building around all of the kind of headacheinducing stuff and but still using SAP. So, like to both of your points earlier, like the these like legacy SAS systems are just like so deeply embedded that the newer insurgents are just coming and they're, you know, building on top and around them rather than kind of like trying to rip out and get people to migrate completely. A lot of what we're seeing AI being used today is for is how do you make it like I think the word is often used like conversational or like how do you pull the information out and actually make it more usable? How do you retrieve the information from SAP without needing to go you know run a SQL query and get all the information or like look at a bunch of screens. It's like okay well if I want to be able to connect to Steven's point around analyze like you know three different uh sets of tables and different geographies like can I quickly you know query that like a natural language way um can I get

reports automatically generated that are customized to me without needing to like go back through the like SAP customization process and I think that is that usability layer is I think indicative of what's happening now with software in general which is accessing the UI is optional and like Going back to the Slackbots point, like you want the information delivered to you versus needing to go to the UI, but the data and the business logic inside either it's SAP or something else that's being replacing it that's building it, but that all still needs to exist one way or another.
>> Mhm.
That's a that's an incredibly important point is for for folks to take away which is that the the the biggest thing about enterprise software is it almost always does what somebody is wants it to do. They just don't know how to make it do that. Like there's no report that that SAP can't generate. No graph, no chart, no analysis or whatever. But you you just can't figure it out. Or maybe it's configured so you don't have permissions or something. So the two the way to think of it is in enterprise software the two most frequently used features exist in no enterprise software natively it's export to Excel and export as CSV
and or PDF you pick and so all enterprise software the first thing they have to do that's missing when they show up and they do that first demo is the customer say does it do export to Excel or does it do export to CSV or PDF because then you know you have an escape valve to do the thing that you couldn't

do before on analysis.
And and so what's so cool about where we are today is that now with with the language models you have this incredible way to actually consume those in much easier than you could before. If you think about PDF, like the old way used to be like, okay, I want to figure out like exception handling, some report that that my system emits, you know, declined expense reports or whatever, but I want to do it over some weird time period or across different currencies that it doesn't handle or some weirdness that you can't figure out the UI. So now you can export them all and take this 20 PDFs and put them in a model and do a bunch of analysis that you couldn't do before or if you did it was all copy and paste and this mundane thing and to something that Sema wrote about these sort of I don't remember the word you used but these ad hoc business processes are the ones that really become the most interesting because they're interesting because that's how a business runs but they're also interesting because those are the next products
Like those are the next companies that people start. You know, CRM used to just be a spreadsheet.
Like the this if if you were in a business and you were account manager and you kept track of your accounts, you just kept track of it in Excel and then a company got started to to do that. It wasn't SA. It wasn't Salesforce first.
It was the predecessor called Seible.
And and then like people like oh we should make a whole company that does this. And that's what's that's what some of these apps are that you're seeing

using language models and interfaces
that are chat to SAP or to Salesforce.
They they are just trying to take
advantage of what the LMS are really
good at which is synthesizing and
orchestrating unstructured information.
>> Yeah. I think we also forget that like
Salesforce is like really an enforcement
mechanism for like the the the go to
market team which is like okay are you
collecting all of the information that
you need to and of course we can talk
about Salesforce hygiene etc as a
separate point but like do you have all
you know is the human doing the getting
the data to then have um like the to
then you know capture the state of the
business
>> um which okay but I think if we like now
switch to the agenda world Um, and I
think again we can talk about what agent
means, but imagine there's an agent that
needs to do outbound calling or outbound
messaging. They want to be able to
retrieve that information. They don't
really care about how the fields are
organized or, you know, how many clicks
it takes, but they do want to be a they
still need to access that information.
But then the second piece they need is
this context thing. So, we've talked a
lot. I think I feel like the internet
has talked a lot about context graphs uh
over the last six months, but what is
that? That's all like the exceptions.
What do you do? What do you how do you
handle certain cases? It's the edge
cases and the permissioning and all that
stuff that needs to be um and all the
policies that are not necessarily in the
fields of Salesforce. Um, and so for the
agent to then go go back to this 8020

thing, the agent can, you know, extract all the information, send an outbound email based on the information that's in the CRM around the person and their persona and what they do and all that, but then like, okay, how do you deal with um, a case, one case versus another, and how they respond? And it's like, oh, well, normally if it's a person who's in Asia, we respond this way, but if it's a person in the US, we respond this other way. That's not captured in Salesforce, but that's that's was in someone's head. And so that's the context that I think is really important now for agents to be able to act on behalf of this data.
>> Oh, that's super. I mean, for Salesforce in particular, that's incredibly important because I've never met a saleserson, an account manager, an account executive who thinks that the default is the right answer for anything with their account. and and like no, even if they get the Japanese language right, you know, oh, it's spring and the birds are chirping and but you're you're overdue on your payment, your license count is wrong, that even if you do that correctly, the rep is going to want to handle it in their in their specific way. And and I think that this notion of exception handling is just the root of the challenge with with agents, which is almost everything interesting in an enterprise is an exception.
>> Yes. Yep. like that. Like all the people are about exception handling it. You know, it's basically like spend 15 minutes at McDonald's and watch people start in the kiosk and give up and then go watch what they really want and

they're like, "Well, I wanted a
McFlurry, but I wanted two flavors and
mix them together and that's not in the"
and it's always the exceptions. And
everything about automation in
enterprise is handling exceptions. It it
just is. It's the strangest thing like
but you know enterprise pricing is a
great example like how much is it per
seat? Well, you have to call us. Well,
you call then you talk and then it's
still an exception.
>> Yeah. And that that's exactly right. So
these exceptions and aren't they're not
captured anywhere right now. Mhm. Um now
I think if you say if there's you know a
voice agent that is doing um let's say
compliance check calls for freight as
one of our portfolio companies does
they're now collecting the exceptions
through their voice agent and getting
some of that context or if they're we're
looking we can talk about computer using
agents if you're observing humans and
how they are clicking through software
how they are responding to things and
now we have this ability now to you know
not only record data or like record
interactions but then process that BLMs
then you're able to start collecting
some of this context but it's a lot
there I mean as Stephen was saying it's
ever all the interesting work is around
the exceptions so it's not like okay you
know 3 days later we've we've we've got
it all we've got all the context because
also like sales cycles take a long time
each exception isn't like handled um
with the frequency that you get the data
immediately right um and so you have to
feel comfortable you have to get to that
point where you're like okay we've c

we've observed enough interactions to actually capture to understand the exceptions and then on the sales side, the buyer trusts that uh that you know this piece of software can actually you know has captured all the context to handle. Well, let me just add to building on that because I think it helps us to to go back to this notion of headless and what are both the challenges and the opportunities because of course if you're an engineer with almost everyone talking about what's going on in the world in AI today is you think headless and API a agent API just interchange them and so you you think oh well it's code I can write the business process down and and the the problem that you hit right at the beginning is is that if you're not an engineer, you can't even explain the process that you use to resolve a customer issue. You and in fact, it's sort of very interesting to watch Amazon really do some of the best work on this because they really really don't want to have humans. Like they you you you can't call Amazon for anything like it's just hopeless. And so what they're doing is they're learning with everybody like the best way to automate something and it's their religion. And it's their core principle which is you just decide it in favor of the customer. You know, oh they sent the wrong thing. So you go to the chatbot, you tell them. The chatbot understands that you got the wrong thing and it just sends you a new one. And I think that that's so interesting compared to sort of old school exception handling. And then they use the data to go and improve the internal shipping and handling and

warehouse process. Maybe it's the product description, a zillion other things or reviews. And and so I find that's what I find so interesting about the capabilities of AI is that it's driving a different definition and different behavior at companies about how to handle exceptions. And I think when we get through sort of the 1.0 version of this. We're going to get to a new version where people are like comfortable letting AI do or decide things because they realize it's adding a level of predictability and repeatability to their enterprise. It is. It's funny to think that maybe customer service gets worse in the short term because things stop getting default decided in favor of the customer, you know, like suddenly suddenly you actually have to defend your case again instead of like you being, you know, re-shipped the uh the sensitine toothpaste like you you feel like you were owed. Um, but uh I think I I think maybe then more generally it it sounds like you're both saying that automating the long tail is still kind of the hardest thing about about all of this. Is is that true or would you say that there are other hard things uh that that developers and founders also need to think about? I think that's that's part of it. Um I think there are a lot of other things around like permissioning and that this is this is all I think I could you could probably lump it into the hard tail but oh sorry long tail but like permissioning is part of this right and like you know as you give people or give it you know API access it's like okay so which in which cases can people

extract data when when can they write versus read like that all needs to be figured out over time as well and like interactions also between agents right um and I think if you go back to the idea of a system of record there's it's one ideally one set of one central repository of data that is the the source of truth well that now if you're have multiple people accessing and writing to it like who gets to access when and I think that's a that anyways these are additional problems that I think need to be solved solvable but will take time you know
>> well one of the things that happens in in technology shifts is it you know everybody knows the thing about nobody understands exponential when it's happening so you have to be very careful to extrapolate and end up extrapolating linear when something exponential is happening. But the same thing happens with productivity or an analogous thing happens with productivity which is people look at the existing body of work that happens today and they say okay how do we make that easier and then all of a sudden there's all this fear that we're going to automate everything away that everything is just going to become an API which developers and engineers say oh that will be easy and then we'll be in this nirvana world where everything is automated and easy and predictable but they forget that productivity drives new scenarios. And so the minute that you can get something easier with automation and you can actually automate it, which I do think is happening right now with agents and with with language models, well then we're going to dream

up a whole bunch of new stuff to do. Like I I just mentioned this this loop that Amazon must be in on customer service. Well, they got rid of all the phone people and the phone experience that would be miserable to do a return and the the challenge response and the fighting and like, can I return this and do I have to package it up or will you just ignore like toothpaste? They don't want it back. Like that's an a pioneering invention by Amazon is like, you know, if somebody gets the wrong consumable, we just don't want it. Like they're poisoning it, they used part of it, it's cheaper to just have them throw it away. Well, that never happened before. Like, you used to have to actually bring spoiled food to the supermarket and show it to them. And so, they've fixed that level of productivity, but now there's this backend that's just out there constantly figuring out how to have it not happen again. And that now they need a new level of analysis, a new set of tools. And the long tail got no shorter. It just got longer in a different way.
>> Yes. And and I I think people forget that that's how innovation is this constant reinvention and it's it's a growing pie, not a static pie. And and all the negativity around AI comes from just thinking that the work to be done is this fixed thing that takes n people and m amount of software and we're just going to replace n people with m plus five and then we don't we're done. There's no jobs anymore. There's just an agent running. And that's just never going to happen. Like legal is a great example of this, like where people do

contracts and they think that the law is going to help contracts get get done quicker without lawyers. Except I can assure you contracts will get longer and more sophisticated and encompass way more sets of scenarios than a person ever could. And that's going to create a whole
>> more litigation around it. And that creates a whole ecos.
Look, there's the the now apocryphal, semi-apocalypical famous example of radiology, which is a correlation, not a causation, but radiologists all love AI, and now we are having a radiology shortage. It it's not it's there's a lot of reasons. It's complicated, but it it it it just shows that the innovation wasn't static and and the market for the demand wasn't static. And and so I think that a lot of what happens just in the micro at the enterprise level is the minute you automate the most mundane thing and think you have it all squared away whole new things appear like actually like expense reporting is a really good example. You know, first there's nothing, then people figure out how to like do spreadsheets, and then people figure out like, oh, now we have a whole system. We can analyze it. And now all of a sudden business travel, you get ahead of the curve. And you're like, well, now let's just use miles for business travel. Let's, you know, route our travel requests to the best prices we could get at any given moment rather than just default to one carrier. Let's use a specific credit card for business travel that buys us a bunch of different added benefits that we know matter to our patterns of travel. And so suddenly

like there's a bigger job called business travel analysis that takes way more people than just booking the flights which everybody can just do on their own.
>> There there's always another layer of analysis on top.
>> Always there. But the analysis then drives new processes and new behaviors that themselves differentiate companies. Look, business travel is a to stick with that example is a huge sync in most companies. It's just a giant expense hole that they wish they could shrink. But once they can tie it to how things perform in their company, then it's more than just expense moderation. It's actually figuring out performance optimization. and figuring that whole thing out becomes like a different kind of job than just booking travel and analyzing expenses. It just becomes this whole remote work optimization tool and then it's a different thing. I think the other thing interesting not to double and you know to spend too much time on business travel but I think it it also ties there are the physical and like digital worlds like there are always things that there will be humans doing you know maybe it's not back office TPS reports but like sales people will be closing deal there will be human interaction to close deals um people will be getting on planes as a result and uh maybe they aren't spending as much time entering data into Salesforce or doing things along the way, but they will there there will be these humans doing online and offline world work and I think that actually is something that you know there will always be a data

exhaust from things to capture optimization that needs to happen and that that isn't going away either.
>> Yeah. Well, I think open source software development is actually a really good example of this because like the hardest thing in software development is, you know, you have to be finished at some point so that everybody knows this is a stable release and can go build on it. And the art of finishing is this long tale of like not changing the code. And and there's no API for that. Like developers think there don't they developers don't hesitate to think there should be no API for that. They they they could think of a way to automate it with voting and with a discussion that has sentiment analysis or whatever, but they you still need a bunch of people to concur over a decision to fix or not fix something. And yet they'll adv those those same people will just say some other business process like closing the books for earnings that should just be an API.
And it's actually the literally the same mental model. like there's a bunch of stuff and we're deciding when to close the books and what sales to account for what and where. It it's fixing a bug and there's a story around it, a narrative and we have to explain it to our boss and if something goes wrong, we need a trail that explains who did what and and so so much of what a business really is it are just the people deciding things. And all that software does is it uplevels, abstracts, and changes what they decide and how and what tools they use.
>> The the other sort of followup to Sema's

point is like it's the best case for just recording everything you do like all like just voice recording everything you do to like capture if if people are, you know, going and flying and closing deals in person. Um, make sure the software or the LLM can kind of capture everything that happens at all times. Obviously not advocating for you know full ponopticon but
>> but synthetic gathering.
>> Exactly. Exactly.
>> And it whether it's like you know recording and you know conversations or taking emails and you know written artifacts and ingesting them. This is all that is the way that the world is moving and exactly.
>> So yeah.
>> Yeah. Well, it's also to your earlier point, you know, expertise exists in this cloud in an organization and and it is the the untapped resource of the modern era and Aaron Levy at Box has done the the most eloquent job of explaining repeatedly the the assets that exist in all of these, you know, Word and Excel documents strewn throughout a company. And it's actually very very hard to to understand which documents are important, which ones to believe. And part of being in a company and having a culture is really knowing the answer to that. And and it's super interesting to watch the customers at Box use Box to to to actually answer those questions. You know, which are the sales PowerPoint presentations that are actually working? which are the spreadsheets and the models that people actually rely on and and I think that that AI is the first thing to come along

that really taps into that unstructured information in a company.
>> Yeah, I think um before we wrap up, it might be good to to visit the sort of more immediate history and then the more faraway history of kind of what what headless software even is. Uh I know Stephen you wrote last year a piece uh in reaction to to the rise of MCP servers. Um and in that piece you also related it actually to uh sort of like early Microsoft litigation that the that the Justice Department levied against them and and part of the argument was that Microsoft had a lot of products that could be categorized as middleware. Um, and I just kind of curious, you know, in all of these different software waves that you've witnessed, kind of in in what ways is history rhyming and repeating? Maybe not on the litigation side, but on the, you know, product level. That part will continue, too. Yes.
>> Yeah. It's it's super interesting. I, you know, and I I I love Sema Opine on on where she sees things going with startups in this regard as well, so I'll go quick. But the the real thing with MCP is it's very much like everything we're seeing now is that it so much of it is driven by an engineering view of what would make for a good software architecture and very little of it little of it is being driven by sort of the using seamlessly the physical reality of of the world. And so of course if you're an engineer and you would love to have like every tool you want to use to have a very clean API preferably like a command line interface that pipes text in and out would be

perfect but that turns out to like not be how the world wants to work. There are many many reasons why it doesn't want to work that way. Sema touched on many like security and compliance and things like that, but the reality is is that no software wants to be disintermediated by some other layer above it. Like nobody wants to to just be put in a corner and said your job is to just store the this SQL format for expense reports and do nothing more and then we're going to use you only for that and then by the way we're we're piping you through to some other tool to analyze expense reports because that's a that's not a growing business that's a decaying business and and so it this whole notion of like everybody is going to be perfectly content to be abst racted by some, you know, benign layer in the middle. It it just doesn't really work that way. And and it's because customers actually do not want to assemble their scenario from a bunch of different providers because all it takes is your system will only be as stable as the most unstable part of that. So if expense report company goes out of business, you're you're completely out of luck. So, you want your expense report company to be thriving and doing more stuff even though you in your head you're like, I I wish they would just stop. I don't want any more from them. It's getting complicated. Oh, they just did a UI reworking that's driving me crazy. And the flip side is the those companies, they they're not just going to sit there and decay and and they're going to look to the left and they're going to look to the right and they're

just going to do the stuff that they see people using with their product. And so SAP the example Sema used, we're seeing this whole ecosystem grow up and SAP is just going to do those things and that's the N now not all of them and most of them they're not going to do very well. In fact, just before this I was talking to somebody and we reminded them that in most giant enterprise companies they view just a tie with some competitor as a win because they'll just bundle it into their existing thing and give it away.
And so it's but this middleware layer it's always always very unstable. It looks great in a network hierarchy diagram of the OSI levels of networking but it's just never that stable.
>> Yeah I think two things I'll add. So one is yeah the the practical realities like even go back to the Salesforce example or like workday. Workday has had APIs that you could work with. But can you really actually extract all of the data out of workday in a like clean way and just operate without using workday? No. Workday makes it extremely difficult to actually like get access to the documentation and work with the like and they don't they don't expose all the end. everything about the API to use the API example is this is you know analogous to what we're seeing now which is then it makes it a dumb database right the and so they're not incentivized to do that so I think what we're seeing is there's three paths in for in front of you one if if you were a a consumer or like a business that's looking to buy software one is okay I take Salesforce and I either turn on

agent force or build all my agents on top of it and then treat Salesforce as kind of the the just the back end I think to what we just talked about some of that will work but some of that will also not work because Salesforce doesn't want you to have want that to be you know to they don't want to be just the data in the background right
>> um and so you know I think that that there will be mixed mixed results around that and I I don't have I'm not bullish on uh the incumbent software building great agents on top there's option two which is you just totally DIY it you have the most control in that situation um however I think to everything we just talked uh that's really hard, right? Like you have to rebuilding true enterprise software and I think for a startup building rebuilding a CRM much easier for rebuilding, you know, a CRM for a Fortune 500 business,
>> it's it's a lot of business logic to capture and you're also trying to like do open heart surgery while like the patient is like alive, right? Or you know, whatever one you want to use the analogy.
>> Well, hopefully they're alive, but
>> yeah. Yeah. Yeah. Yes. Yes. Yes. Of course, if they're alive, but I mean you're like taking the engine out mid-flight, whatever you want to say as the analogy. Um that's that's really hard and you have to get the like practical realities of permissioning and collaboration and all that right. Then there's a third option and I think this is why we are continue to do what we do in investing in in AI software is because there is a reason that like you

know agents can continue to be built the data can be sucked in and built in the background. A lot of what we're seeing right now is um things that are working alongside an SAP or a layer of visibility on top that is um enhancing the experience and allowing the business user to then run agents on top of the existing data they have um and not but also not like throw out all of the the logic they've had in the background. Um and then I think also create a new system of record like voice agents are collecting new data recordings are correcting collecting new data transcription ingestion of documents all of that documentation is pulling in and maybe you know one day these AI uh startups will replace the systems of record in the back end but they are doing so in like a systematic way of observing how the business is operating. I guess I guess to close this out though, Sema, you you sort of just touched on this. Where where are we really seeing the biggest opportunities for startups right now? I think look, a lot of this Yeah, what I was just saying, it's it's it's doing the things that the incumbents are not doing right now, which is um which is going from a layer of collection of data and into how do we take action on top of it, right? And so um take the CRM example, right? It's like I'm not just logging all of the like call information but then now I'm providing the intelligence back around okay how do I prioritize leads which uh accounts should we work on what have what has risk of churn flagging all of that and then like sending the outbound right and so and and and part

of that is creating this agentic loop which is you now as the agent sends the outbound sees the response you're understanding okay a what works what didn't what how did people respond and then b you're also collecting like benchmark data view on like okay this type of response is most effective in these cases and in Asia we should be using this language you know type of opening versus in Europe etc that sort of stuff you're now agentically collecting all of that um and that's like an interesting data exhaust so I think that's another area and the third area I just would flag too is we talked about this like physical realities but the other part of physical realities is a lot of the vertical software that builds for the you know the physical world actually um and that is a really interesting interesting set of data that's not it's like hard to capture has been hard to capture historically. Um and you know you will have to continue to um pull together things that um can be cap you know agents have been able to operate on software but then also what humans are doing out in the field machines are doing out in the field and pulling that back in. So like construction, manufacturing, all of that. Well, the universal truth for enterprise software is the the most difficult thing to do that happens to be the dumbest is to attempt to just compete head-on with with an existing category. And and by head-on, I mean not just the same category, but doing it the same way. The biggest opportunity right now is always always to look at the existing

uh sort of mental map of enterprise categories and be in between two established players because the thing that you know right now during a massive technology shift is the one thing that established players won't do is disturb their existing product line and go to market. So they absolutely will just be bolting AI on top of their existing product. They they won't be getting rid of it. They won't stop working on it. They won't do anything to to break it. They're just going to try to weather this technology storm by sort of power throughing it, powering powering through it. And so your opportunity in a startup is to just look at two big players who are bolting AI onto the side and exposing some existing API as an agent or whatever and just aim for the middle and do things in in the new way and the new way. And by not attacking head-on, you don't show up at every single customer and have them go, you know, well, you need to the do these 8,000 things before you even enter the door. Instead, you have a equally difficult question, but one you're in control of. It's which is why do you even exist? And and that but you that's your own question. You you don't have to answer to a a series of 20-year frameworks that a 20-year-old framework that got created to answer a bunch of questions that aren't even relevant anymore. And and the best example of this is is is HTTP and HTML. client server existed, but the reason that those took over was not because it did all the things that client server did. In fact, it did none of them, but it implemented that concept in an entirely new way. And so, the web

exists in spite of the fact that legacy vendors had a trillion dollars invested on how client servers should work.
>> Well, and I and I would say the other piece too, it's not just two between two legacy vendors, but I think now there's like a layer of translation between two different functions within an organization, too. Oh yeah. Yeah, for sure.
>> Software has always sold to like oh I'm selling into just you know the sales team or the finance team but then there's like these handoffs and which is now the context right but on bills and deals and like that actually also presents an interesting opportunity. So the last question I have which is for Stephen is um so network effects is this thing we always talk about on the consumer side and it's a great you know source of defensibility. No enterprise software business as far as I can tell has successfully done you know implemented network effects but you could argue that is a good source of um durability over time right and I think Salesforce has tried this in in in in a couple ways in the past but do you think that like enterprise software will start entering the the uh the you know the field of network effects in terms of like okay we're going to have both buyers and sellers on our CRM and therefore be able to like mediate these transactions or like yeah I'm curious to get your take on that?
>> Well, certainly network effects outside of a company are extremely difficult for a bunch of compliance and security reasons,
>> but but the biggest network effect in

enterprise software is inside of a company. And we're seeing that happen now with with just chat. like all of a sudden you're seeing the it's it's so incredible to be at this dynamic that that almost felt like the good old days when some very motivated person like most people who work in enterprises it turns out are not like super interested in making their job better they actually just want to go to work get paid and go home and they don't come to work every day going how can I make my how could I streamline my task they just want to not mess it up that is a lot of the world but There's a small set of people like those bankers at Goldman Sachs that were like, "How do I do more deals faster, better, more clever models?" And so they were using Excel when when the other bankers were using one 123. There's actually floating around on the internet is this old commercial for Excel, the launch this launch TV ad from the from the late 1990s where or sorry the late 1980s where the first Excel spreadsheets were being used and it's a person sitting there with this monstrous laptop that weighed like 12 pounds in an elevator trying to use it. I I'm laughing because of course they were trying to not run out of battery life in the in the elevator ride which was invariably the case. But all all of a sudden this crowded elevator of a bunch of people in in these 1980s ties and 1980s wearing glasses looking at the spreadsheet going, "What are you doing? How are you doing that?" and getting all excited. And fast forward to 2025 and that's exactly what happened with chat. Like, in fact, I had a friend at SAP

that was was writing like a um a SAP white paper about something and and I I just asked them, "Tell me what questions you're trying to answer." And I did the prompt and sent them back a white paper. And I'm positive I kicked off some sort of viral loop, not technically a viral loop, but some sort of network effect viral loop inside of her team because like all of a sudden people are seeing how to make their job better and it's accessible to them and they're doing it. So I I think and to your point Tim like this idea of of a tool that enables two functions to talk together that couldn't before is golden. Like that's exactly like that's literally what enterprise software integration is except that's all manual brute force higher accenture kind of stuff. And so if you have products that bridge this and you know Figma did a bunch of this with design and product development. And so if you can develop software that leverages AI in order to bring together parts of an organization that don't normally communicate that's a whole that's a new category.

And we've seen that with things like IT budgeting where IT and finance would end up with tools that ended up helping them both do forecasting and the cloud enabled that. And so I I think that that's that's a huge opportunity.

>> Nice. Well, I think that's also an amazing note to end on. Uh thank you so much, Stephen, for for joining us here. Thanks, Sema. Yeah. And thank you. And thank you, Sema.