Chris Strahl:

Hi. And welcome to the Design Systems Podcast. This podcast is about the place where design and development overlap. We talk with experts to get their point of view about trends in design, code, and how it relates to the world around us. As always, this podcast is brought to you by Knapsack. Check us out at knapsack.cloud. If you want to get in touch with the show, ask some questions, or generally tell us what you think. Go ahead and tweet us @TheDSPod. We'd love to hear from you.

Hey, this is Chris Strahl, the host of the Design Systems Podcast. I'm here with my friend Chris Deluca. Chris, it's been a really long time since we've had a chance to hang out, so I'm glad we're doing this. You're over at Lullabot now, but before that you were actually at Memorial Sloan Kettering. Which was an early Knapsack customer. And so really stoked to have you on, jam about that a little bit and about just design systems in general.

Chris Deluca:

Yeah, thank you so much for having me on the podcast. It's great to be here.

Chris Strahl:

Yeah. So before we get started, tell everybody a little bit about your background and your role. And maybe a little bit about Lullabot for those not familiar.

Chris Deluca:

Yeah. So my name's Chris Deluca. I'm a Senior Front-End Web Developer. I've worked a lot in media over the years. And yeah, now I'm at Lullabot, which is a digital design agency in one of the first Drupal agencies. Kind of clutch in that field.

Chris Strahl:

Yeah. And funny enough, I happen to know a lot of the bots from my years in the Drupal community. So it's good to see you going to a great place.

Chris Deluca:

Thank you.

Chris Strahl:

So as an engineer and as somebody that has worked a lot on front-end technology, give me just a quick rundown on your design systems experience. How do you think about this stuff? What is it you typically work on? Give me a better sense of your work in this field.

Chris Deluca:

So my first kind of design system experience was at Memorial Sloan Kettering Cancer Center. And we were building out, at least for the beginning, was a design system for their entire mskcc.org marketing site. Which serves doctors, and potential patients, and current patients. And just a wide breadth of people who are interested in the topics. So I had to take a lot of different people with different agendas in mind. But the way I always start with those types of things, with any system and design systems in particular, is to focus in on the people. Because the way that people interact is ultimately how this is

going to go. It's really that soft stuff. The technology is hard, but the people, that's where everything happens with. Who you have to convince, whose life you have to make easier. And so on and so forth.

Chris Strahl:

Yeah. And I actually have a fair amount of history with MSKCC as well, in that when Knapsack was still an agency, when we were Basalt. We worked a lot with y'all around the implementation of what ultimately became the MSKCC design system. And then you guys became a very early Knapsack customer. I think in the first two, if not the first one. And so early believers in us, which by the way, I really appreciate. Because I know you were part of that decision process. But tell me about those early days, because being a part of this foundational story was really interesting. The way I recollect, is that you guys had a bunch of components pretty deeply embedded into your CMS platform. And that was functional from a component standpoint, but really difficult to use in a variety of contexts, like you would want for a design system.

Chris Deluca:

A 100%, yeah. The first thing we were trying to do is just get some brand consistency. Build on small goals. And just even on our own site, forget about distributing to others, we didn't have consistency across our site. When we did an audit, myself and my coworker, Amanda Cove, went through just the whole site and just documented every color in use. Every component, every button style, every carousel, because design would come down. And we worked with a design agency, so there wasn't necessarily a continuum of knowledge between each design session. So things would get repeated. So just finding those duplicates was big.

Chris Strahl:

Yeah. I mean, it'll yield interface inventory. And I think one of the interesting things too about that design partner was they were still thinking in pages. A lot of it was about, what is that final experience? Not, what does that component look like? But all of that did change. You guys did this interface inventory, you got all this stuff collected. And then what happened?

Chris Deluca:

Yeah. So then we have this big library of stuff, but now we need something to do with it. Now we need to actually build a system around it. And so I guess two things were happening. One was that we're messaging upstairs to say, "Hey, this is really valuable. And here's why." And a compelling argument at that time was, "This will save money. We're going to reduce the amount of work that you have to design and develop. That's two verticals that will be less if we can systemize this a bit." And then the other part was actually crunching through all those components, and developing them in a kind of display tool. And that's where we developed the relationship with, what became Knapsack. So working with the Basalt team, at the time, to just go through every component. And there's many. I mean, I think, the final system has over 50 unique components. So a lot to crunch through from a big initial set.

Chris Strahl:

Yeah. I remember there being several hundred. And taking those several hundred and then condensing them down to things you could count in the dozens very easily, was a pretty Herculean effort. And it was actually where we got a lot of the early ideas for how we think about workflows inside of Knapsack. Because there was so much change happening, a truly massive amount of change. And not just across

one website, but across, I want to say seven or eight. And they were all changing simultaneously using that same component library that we were ripping out of Drupal and sticking into this separate abstraction. And that change control, we kind of all decided collectively that Git was our tool for that. That we were going to basically say, the workflow had to go through Git in order for us to be able to maintain control of how the system changed.

Chris Deluca:

Yeah. And once we moved to Knapsack, the versioning was key. That was a big selling point. To be able to have automatic versioning, so then any dependent will know exactly what version they're getting. And you don't have to structure releases. Structuring a release manually can be really valuable for some projects. But like you said, the change was happening at such a pace that we needed a way to automate that. And that's what Knapsack delivered. So that was really key.

Chris Strahl:

We were doing majors weekly. Or sometimes multiple times a week. And so, when you have that breaking changes happening all the time, the ability to know that as a product owner, I don't have to update everything about my package dependencies every single week whenever there's a new major. I can still stick with something at least for a couple more versions until I'm ready, I guess.

Chris Deluca:

Right. Yeah. Having that freedom to not break everyone is huge.

Chris Strahl:

Tell me more about what this ended up achieving there. Because I think that, in my mind, where it ended up was this place where you all were able to do a lot more with a lot less. And I'd kind of love you to hear your take on that.

Chris Deluca:

I think that's exactly right. So I was in the marketing team and we worked on the public facing websites. And we're not large. I think we had maybe five developers at our peak. So for the amount of sites or the amount of surface area, hundreds of thousands of pages, that's a small team to address all that. So having a single place that we can develop those changes and iterate quickly. And show stakeholders and QA, in a kind of isolated environment, that really increased our velocity. Or we were able to meet the velocity that we were expected to achieve.

Chris Strahl:

I would say that's not uncommon now either. You all were in a very interesting situation. It was the first time that somebody had put a flag in the ground and said, "This is going to be a call to design system." It was the first time this thing was given a name. But it's not like you all weren't using components. And it also isn't like there weren't other systems similar to this inside of your ecosystem. It was just at this one moment where you had this, I don't know, "not really greenfield, but also not really brownfield" approach to a design system. Maybe a patchy, tan-ish field that hasn't been watered all that well. And so I think that it does mirror a lot of more modern experiences, as well where people are taking, either multiple design systems and collapsing them together into a more singular system. Or they're really

looking at this idea of, "Hey, my roadmap is still really, really big. And without more resources or a different way of working, it's going to be really hard for me to attain that."

And especially in today's market. That, more resources thing, may not be an option for a lot of folks. So do you often relate that now in your work with Lullabot and more on the agency side? Do you relate that experience to some of the stuff you're working on?

Chris Deluca:

Yeah, just having that way of stretching your brain to think about systems. I think it's often on the front-end at least you can not think of things as a system. I mean are everything's a system, but it's possible to just like, "Okay, this is this one blob, this is other blob." And it'll work, up to a point. But once you hit a certain scale and a certain velocity, it doesn't. It falls apart. And you're playing whack-a-mole. And it's that classic problem where you're just constantly putting out fires that you've created, essentially. So whether or not the project is called a design system or not, it's all systemic thinking throughout. How does this relate to something else? I think of it almost like as, trying to think about everything as a component, but then think about each component as a series of relationships. It's not just the thing itself in isolation, but it's how they relate to other things. And how they're going to synthesize it, integrate.

And trying to keep them as modular as possible. Just a simple example of, okay, maybe we say, let's not put any white space around the component. And we let the containing component add the white space. So, button doesn't get any white space. The containing component gives it the white space.

Chris Strahl:

And that sort of idea of thinking about a component in context is something I've been building around with a lot in my head. Trying to figure out the right description for this. Because we oftentimes still think about the web as a visual medium. We're all drawing boxes on the internet, and then filling those boxes with words and pictures. And so something is to the left of something else. Or something else is centered vertically on the side of some other place. But the reality is, is the web is much more fluid than that. And it more resembles a bullet list with an order and a depth. And an awareness of that order and depth with those components. So every line in that bullet list is a component. But it's also somewhat misleading, in that there's a lot of data that goes into that component that also is aware of all the other components around it.

And so I've been wrestling with a much more, sort of visual metaphor for this. What design is helping us, say, visually is like, I'm showing you a picture of a cake. And now it's up to engineering to figure out how to bake that cake. And figure out all the ingredients, and everything like that, that go into it. And that cake visually looks like something that you would see in a display window or whatever. But that recipe is a very structured set of instructions about how you get there. And that to me is the fundamental difference in the way that we used to think about the web. And the way we're starting to think about it now, is the idea of structured data matters so much more than it ever has before.

And that structured data and how it relates to other structured data, represents the fundamental way that we build. And without a component model, or if you don't think about things in a systematic way, you sort of lose the plot in that that structured data becomes irrelevant. And now there's a bunch of things that, it's like a bag of doorknobs. You can't really do much with it.

Chris Deluca:

Right. It's literally impossible to output structured data if you do not have a structured input. The easiest way to do that is just structure it how you want it to come out on the opposite end. In a somewhat similar way, I'm thinking about chatbots and AI and all that stuff.

Chris Strahl:

Me too. It's literally where my brain is.

Chris Deluca:

Yeah. They feed on structured data. I mean, I've heard some people musing about like, "Oh, are we going to be just writing for chatbots? Is that going to be the primary audience for the web?" I don't know. But it's certainly a audience.

Chris Strahl:

There's a couple of interesting things there. Yes, AI completely feeds on structured data. And I think that a design system is an interesting input or constraint to that structured data. Use components to build a thing. Or build this thing using components is maybe how that prompt would get structured. But there's a whole other side of this that is definitely the idea of, how aware of systems are the AI responses? And so you talk about a human that sort of loses the plot and fails to think about systems and how components interact with other components. Does AI have that same problem? Is it vulnerable to that same weakness? And that's something that, I think, it remains TBD. And I think that in some ways it is. And that I have yet to see something that takes a visual design and represents it as code I would put into production.

I've seen lots of things that will make an approximation. That will say, "Here's some HTML that looks like the right thing at the right page with the right viewport size. With the right browser. With assistive technology disabled." And if you don't look too closely or have more than one user looking at it. But I've yet to see something that I would say, "Yeah, you go straight to production with that." But I've seen lots of things in design systems assembled by people, that when you look at the output of it, you're like, "Yeah, that's professional code." And so I wonder, are we looking at an AI-enabled future that helps people build using components? Or are we looking at one, where visual design to code is easier? And I tend to think the first is what matters more. And I'd be curious your opinion on this.

Chris Deluca:

That really is interesting. I like that. Because I too have not seen chatbots generate code that I would ship. And they could obviously get better, but I think they'd have to get pretty leagues better. Look, I'm no AI expert, but from how I understand how the process works, it's just hoovering up data that already exists. So it can't come up with new ideas. It can only say something that's similar to things that someone else has already said. And probably towards the median. So not towards the high end necessarily. It's just kind of an average. Whereas if you have, let's say crafted components, a human crafted components I'll say. And then have an AI to help you assemble them, that to me hits the sweet spot of where maybe you're leveraging what each is good at. I know AIs are really good now at doing transcriptions. So that's taking just this kind of messy audio thing. You don't know the different tones of voice, all this kind of noise in there. And then putting out, "Here's exactly what they said. Here's the typed form."

And to my thinking, that's not all that dissimilar from taking the messy input of a human and saying, "Oh, I kind of want a page like this." But using structured components to kind of spit that out in a more structured way.

Chris Strahl:

Yeah. It gets to your earlier point of, to have a structured output, you need a structured input. And the idea of an innately unstructured medium, like a Figma file, becoming properly structured in ordered data that represents a webpage, it's a big leap. I'm not saying it's impossible. I think that it absolutely is possible someday. But even with all the brilliance and power of the LLM/AI revolution, I still am in a situation where I can't find something that would be output from a unstructured medium that I would place in production at scale. And until I can see that, I think that we are still in the age of component assembly, not visual design to code. And I think that it's going to be kind of a bit of a race to see if those visual to code tools end up outpacing the "I can build with code, but I never have to write code" kind of tools.

Chris Deluca:

Yeah, that'll be interesting to see what kind of comes out on top or neither, I don't know.

Chris Strahl:

Yeah. It's probably some medium or fusion. But the code tools feel like they have a little bit of an advantage, because you're starting with structured data.

Chris Deluca:

Yeah. Yeah. I'm also interested, what you brought up earlier kind of made me think of, talking about Figma. And how we can represent our designs there and structure it in a certain way. But something that we always struggled with, and I think still doesn't have a great solution for, is source of truth. Where do you look to see, this is exactly what we should be shipping? And we can say, "Yeah, the code is the source of truth." But it's always going to lag behind the Figma design, because the designer isn't directly editing it. But maybe that's influenced by these new code tools. I don't know. I'm curious your thoughts on those.

Chris Strahl:

Yeah, it's funny. I was just talking to Kendall from Red Hat, now Pinata, earlier today about this exact thing. Where she was asking about what the future of design looks like. And I think that there is some sort of new role here. Where the traditional production design work of, "Here's a comp, give me five variations of it." Or "Here's the component, give me seven variations of this. And figure out how to store it in some library inside of Figma." I think that's going away in the short-term. And we can feel that presently right now. Where people aren't maintaining big sticker sheets or big UI kits anymore. That's much less fashionable than it was even two years ago. And so the erosion of that is interesting. Because I think that it speaks to this broader trend of, people are realizing that a lot of the work that we do in design tools just frankly is difficult to capture the value of.

And that's not to say the designers aren't valuable. I think the designers are super valuable. I think that they've been sort of typecast in this role of, "I push boxes and fill them with text and images in Figma to make comps." And the reality is, is the web is so much more of a rich medium outside of Figma than it is

inside of Figma. The shit you can do with HTML is way cooler than what you can model inside of Figma. And as such, a lot of design decisions exist in engineering, that I think that if designers felt empowered to take part in that decision process, they would want to. So I think that that's fundamentally the reason why design tools as they exist today, which are largely visual design tools, end up being usurped by new experiences that represent user-friendly and design-friendly ways of manipulating code without actually having to write a bunch of code in IDE.

Chris Deluca:

Yeah, I actually completely agree with that. I just had an experience recently. I mean it wasn't with a tool. But the limitation of a Figma design, even though it uses web technologies under the hood, is that it's ultimately a static picture. And the web is the opposite of that. It is a static picture, every pixel of every device that could ever access it. So it's completely mutable. And so that's just a mentality. It's not the designer's fault, it's just that that's the tool they're given. So that's how they can think until provided with something else. So I kind of proposed instead of doing break points to have a component that, "Okay, now we have this much space in between the image and the text block. And now it's 768 pixels wide. Now we chunk down to this smaller amount of space." Because one pixel above that breakpoint, it's probably going to look a little funky.

We have great tools in CSS now. We can use the clamp function, we can use min, max and all these great tools that allow us to do scaling amounts of space based on how big the screen width is, without using something like a breakpoint. Or it can be smooth scaling. And then we can keep the relationship between the two elements and scale that.

Chris Strahl:

There's a whole constructability thing of modern browser technology in HTML and CSS that, there is no analog for that in a design space. You're not in a situation where elements in a design tool are going to scale dynamically the same way they do in the web. If I make something bigger by 50%, all the elements in the page get bigger by 50%. It doesn't necessarily work that way on the web. And so it's one of those situations where we rely so heavily on visual approximations to do our design work that are frankly unnecessary. And the problem is the process. I still need to be able to run a design process in a code-based tool. And if I can run a design process in a code-based tool that the designer is comfortable with. Look, don't redefine your job, just redefine the medium you're doing it in.

That's the unlock there. And so, I mean, that's what we're working on at Knapsack, is we're racing towards this idea of, how do you have a design process that is fundamentally rooted in production code? And then there's no handoff. And then there's also a bunch of decisions that designers never had access to before that they now have access to. And then there's also a bunch of people in the content and product side of things that can just make things. And I think that that's the really, really awesome power of these AI-enabled tools, is it turns each and every one of us into a creative. Because you don't need to have the really deep intricate knowledge of how a dot map function is better than a, what is it a while loop that it replaces or something like that, right? I don't know, I'm probably getting that wrong. I haven't been [inaudible 00:23:14] in a long time. But one of them is a more efficient function, but you wouldn't know that unless you spent time developing in JavaScript for the web.

And you're going to have a whole bunch of people that want to interact with these systems that haven't spent that time in that technology. And just the mere fact that they could take advantage of that work without having to know exactly the decision process behind it. That's cool. I think it's really cool.

Chris Deluca:

That's really fascinating. Because how that code gets written, by hand, is... A big consideration is, there's going to be a human reading this. Which I think is probably always going to be true. But if there's a lot more automation in those tools, maybe you're not necessarily writing code to be read by a human. So maybe that while loop is more efficient than the map. But it's harder to read for a human, maybe it is a while loop. It's kind of like a compiler in a way. But just upfront.

Chris Strahl:

That opens up all kinds of questions about, is prompt engineering the next coding language? Just all this really big esoteric philosophical things that are out there. But I think that fundamentally the interesting thing here is this idea that it's almost like a new creative channel for people, that is pretty undefined right now. And I don't know exactly what the experience looks like. We're working on trying to figure that out. But I think there is an experience to be had here that flips this industry on its head. And basically says "There is no such thing as a separate design tool from the code that you're putting in production." And that'd be really cool. That'd be an interesting place to live.

Chris Deluca:

I agree. I mean, because I think a lot of the friction comes between that translation layer between design and development. And just getting those expectations that the designer is communicating implemented in real code that makes sense. And squaring those expectations with what actually happens is a process. And right now it's a manual process, but basically offloading or moving that process closer to the designer where they can play with the tools or the same medium that the developers are playing with. If we were talking about this in terms of traditional art, developers are working in granite and the designer is working in oil paints. They're two different mediums. But yet we have to kind of synthesize this. Like, "Oh, I didn't expect the perspective to be this way." It was like, "Well now it's three-dimensional." And, "Oh, the stone doesn't cut this way." There's all these challenges that come from that. And yeah, I think that is the next frontier of, how do we solve those a little bit better.

Chris Strahl:

Right. Yeah. And I mean there's so much of this that it relates to communication and empathy. And before the show, you were talking about Fred Brooks and Mythical Man-Month, and everything like that. And you'd said that there was one thing that struck you about it that was really interesting. About how the mirroring of organizational design and communication flow. I think that that has a lot to do with what we're talking about here. A mutual friend of ours, Brad Wade, coined the term that I use really frequently of the designer engineer handoff as the devil's tennis match.

He's a big tennis player.

Chris Deluca:

He is.

Chris Strahl:

And so he talks about things in tennis metaphors often. But he talked about how, you have this idea of what is truth. And truth is the ball. And you sort of bat it back and forth between design and engineering. And how it ends up being this really painful thing that's almost competitive between the

two parties. And the idea is, is that it really should be something where people are coming together, and meeting, and talking, and figuring things out. And having a communication flow that is very cohesive. But because of the structure of the design and engineering relationship, that communication flow feels strained and competitive more often than not.

Chris Deluca:

Yeah. And it doesn't need to be that way. I think it's there because there is a natural communication friction. And so any type of friction breeds more friction. Actually, I'm curious. So we talk a lot about tokens, design tokens. And I'm wondering if you have any experience with making tokens, like really expanding the role of tokens. And making them, I mean, it's hard to systemize it at a big scale, but to have flexible tokens that maybe we can say, "Oh, this scales between this amount of space and that amount of space." Things like that, where you're really kind of baking in some of the cool tools that developers get access to that designers don't.

Chris Strahl:

Yeah. And talk about something that you started to see designers really bandwagon on. People are like, "Figma give me tokens." And basically begging for it. And so, what's been cool, is to see a bunch of people in the design world be like, "Fuck yeah, tokens." And I mean that is innately talking about key value pairs in code. And I think that that's awesome. And it shows sort of the thirst for these tools that are more interoperable. But in terms of, can tokens do more? All right, so there's two camps here. There's the, tokens are functions camp. Which is basically, I want to have a bunch of reusable utility functions that exist that would be this halfway point between the variable key value pairs they are now, and something like utility classes.

And then you have a bunch of other people that are like, "No, they should never be those things. They should always be variables, or collections of variables as sets. And we should be able to represent our UI just using tokens." Which is kind of an interesting concept. Where if you think about, what is a button? A button is a grid, some spacing, a fill, a border radius, some text tokens. Maybe an animation or transition token. You could represent a button in base elements if you tried hard enough using just tokens. Now I'm not sure anybody would want to read that code or actually use that, but it is theoretically possible. And then you could start to think about, "All right, what tools abstract that to make that easier to manipulate and work with?" And I'm not sure where I come down yet. I know where Evan comes down, my co-founder. Is that, "No, these are key value pairs. And we should be able to figure out how to have some meta around them that makes sense."

How do we group them? And how do we create sets of them? And how do we do typing with them? And all this other stuff like that. But they should never be functions. Those should be higher order components. Which would then be components in your component library that then do something, but don't actually have any UI to them. I don't know. I'm curious what you think about that. It's something that's a bit of a new topic for me. I read Sarrah Vesselov's book. I've got it behind me somewhere. I don't remember the name. Oh, Building Design Systems. Yeah, it's Taurie Davis and Sarrah Vesselov. Where they sort of propose this idea of, everything is constructable down to the token level. And I kind of thought that was a little bit crazy when I first read it. But I don't know, maybe it's not.

Chris Deluca:

Yeah, that sounds very optimistic to my ear. You can structure everything perfectly. And in my experience, the world, and organizations, and people are more messy than that. If you could get

agreement up and down the organization that, "This is our structure and then we're going to build from that." That's amazing. I think that's a fascinating place that I've never, ever seen.

Chris Strahl:

And to be fair, I've never seen it either. Seriously, I've never seen that in implementation. But it's an interesting experiment.

Chris Deluca:

Right. I think it's a good as a thought experiment. Because while I don't think that you can actually get to that place, I do think that it's useful to think about things that way. And to kind of structure as much as you can. But to also be realistic about, "Okay, I don't want to force structure where, as I can tell in my organization, no structure is going to be upheld here." Adding that structure is the over-engineering problem, where now you have an extra problem to deal with when you need to change it.

Chris Strahl:

Right. And your system becomes so complicated that people don't understand it. It becomes brittle because there's only a few people that can maintain it. And ultimately people then don't adopt or use it, because it's too complex.

Chris Deluca:

Yeah, I think there's something big to be said about going a bit under your intelligence. It's kind of a different way of saying, "Don't be too clever." But there's a big value in simplicity. And maybe things are a little obvious, but lots of times that's more durable, I think

Chris Strahl:

It's at least easier for people to wrap their heads around and understand how they can use it. I think that that's been one of the interesting things about design systems. Look, I do a lot of pitching. A big part of my job as CEO of a startup company is to go raise money. What's been interesting is in industry, I almost never have the conversation about what is a design system anymore. But when I go pitch VCs, I still very frequently have the conversation about, what is a design system. And it's also still incredibly hard to articulate in a short sound bite. I don't know if any VC's listen to this podcast. Probably. But VC's think in sound bites. That's kind of how it works. They want some nice encapsulated thing that they can say that, "This is what this company does." Or, "This is what this concept or this category is all about."

And I still find that difficult to this day in design systems. And a systematic approach to building products that involves both design and code, is the best thing that I've come up with. And of course it involves product and content too, and there's a bunch of omissions in that statement. But the point that I'm trying to make is that, when it comes to the understanding of something conceptually, the ability to create a very simple explanation for what it is and why it works, is pretty essential to getting outsiders to like it and to buy into it. And you know who those outsiders are for developers and designers, or executives that hold the purse strings. So if you can't articulate the value of your system in a couple of simple sentences, and why it works, it's pretty hard to get people to buy into that sort of stuff.

Chris Deluca:

Yeah, I would agree with that a 100%. I think it's a challenge that faces any system. That's my hot take. If you're doing a system of any kind, it's really hard to describe simply. Because systems are complex and they're abstract. And human brains don't like abstraction. We're not built for this. The fact that we're actually engineering code at all is kind of amazing. But for a long time, I was trying to be really accurate with my descriptions in a pitch. Like, "Oh, this is what this is. A design system is, well, let me break it down." That's not what anyone's interested in, as I learned. And what was more valuable to me in being able to do that, was to maybe get 80% accurate. Maybe even 70%. As long as it's close enough, but still short and kind of descriptive. Because usually the people you're pitching to aren't going to directly use it. They'll experience it later. But they're not going to actually be involved in that.

And really, whoever you're talking to, if you're just describing something at a high level, it's okay to not be exactly accurate. I tell my engineer brain.

Speaker 1:

Any simple statement has a lot of omission, right?

Chris Deluca:

Yeah. Yeah.

Speaker 1:

There's no way to perfectly encapsulate everything that a system entails in a couple of sentences. I mean, what's fun is, now that everybody at Knapsack is all hip to the systems thing. We have our own design system that manages a platform that makes design systems for companies that then consume design systems. It gets very meta. And so I get to troll our engineers by basically naming components whatever I want inside of our system. And then writing descriptions that are whatever I want. So our description for our button is like, a button, or a CTA, or a link. Whatever you want to call it.

Chris Deluca:

Yeah. I like that.

Speaker 1:

I'm curious, with the jump to agency, and now spending a lot of time with a lot of different folks, as you're apt to do inside of agency. What are the big takeaways from that crazy design systems project that you had at MSK, that you find yourself constantly thinking back to or referring to in your agency work now?

Chris Deluca:

Yeah. So I feel like just flooding with lessons. But I mean, some of them were just, build it to delete it. I think as engineers, we want to build something that lasts forever. We're going to engineer it perfectly and get everything there. If that's at the component level, if that's at the system level. I know I'm definitely guilty of that.

Speaker 1:

Isn't that another Brooks-ism? The build one to throw away? Yeah.

Chris Deluca:

Yeah, yeah. Right. The prototyping phase. Yeah. I guess I'm taking it one step farther, and apologies to Mr. Brooks if he's already said this. He probably has. But just, at least in the marketing side of things or in that kind of space, there's so much churn of change that sometimes it's just better to throw it away. This component is no longer useful. It's more work to reorganize everything to make this have a continuity. The ship of Theseus, it's a new ship now, just rewrite it. So thinking about that in the initial designs. Again, this is hard to kind of state in simple terms, but it does change how you code things. And I think it makes it a little bit easier when, inevitably, there is a change.

Speaker 1:

You know, what's interesting is oftentimes when we think about what represents a really healthy design system, it's that the system doesn't grow much. It's that most of the stuff that's in it, is what's needed. And what that means is not that new patterns don't get added to the system, it's that as new patterns get added, others get culled. And I think that's really hard. Because there's a lot of an emotional attachment to that, right? I put work into that grid of heroes that I made. It doesn't matter that nobody really uses a grid of heroes.

Chris Deluca:

You're right. Yes.

Speaker 1:

But I put work into that dammit. How do we make it so there's a grid of heroes somewhere in our product? I think that there is a little bit of letting go that is a part of building a healthy system.

Chris Deluca:

A 100%. I think that's true probably of every system, but especially on the web where things are fast-paced. I like to think of all my work as a Tibetan sand painting. You put all this painstaking effort into it. You get it just right, and then you sweep it away. And you do the next one.

Speaker 1:

That's great. Of the impermanence of all things.

Chris Deluca:

That's right.

Speaker 1:

Well, hey, thanks Chris. I really appreciate you being on. This has been super fun.

Chris Deluca:

Thank you for having me. It's been a blast.

Chris Strahl:

Let's chat again. I wanted to have an entire conversation about when you get rid of things now. Like, how do you Marie Kondo your design system?

Chris Deluca:

Great. I'd love to.

Chris Strahl:

This has been the Design System Podcast. I'm Chris. Have a great day, everybody.

That's all for today. This has been another episode of the Design Systems Podcast. Thanks for listening. If you have any questions or a topic you'd like to know more about, find us on Twitter @TheDSPod. We'd love to hear from you with show ideas, recommendations, questions, or comments. As always, this pod is brought to you by Knapsack. You can check us out at knapsack.cloud. Have a great day.