Jessica Bell:
This is ACM Bytecast, a podcast series from the Association for Computing Machinery, the world's largest educational and scientific computing society. We talk to researchers, practitioners and innovators, who are all at the intersection of computing, research, and practice. They share their experiences, the lessons they've learned, and their own visions for the future of computing. I'm your host, Jessica Bell.

Jessica Bell:
Today, I have the great pleasure of talking to Radia Perlman, a computer scientist and researcher, who made great strides in the field of networking. You might have heard of STP, or spanning tree protocol. Radio wrote the algorithm which powers this technology. Welcome, Radia.

Radia Perlman:
Hello, Jessica. Pleased to be here.

Jessica Bell:
I'm so happy to have you too. I'd like to start with just a little introduction of yourself, and tell us a bit about who you are.

Radia Perlman:
Well, I work at Dell Technologies. I design network and security things.

Jessica Bell:
Great. So somebody who came into the computing society fairly recently, I've read a lot about your career. And it seems like you weren't necessarily destined to be a computer scientist. I've even seen some interviews where you self describe as hating computers. I'd love to know a little bit more about how you not only got into this career, but ended up developing some of the pioneering technology that we all use today. Could you walk me through how you got into tech and how you got over "hating computers"?

Radia Perlman:
Yeah. Well, I'm not sure I've ever gotten over hating computers. But yeah, I was always good at science and math. I also was good at creative writing and stuff. But I was obsessed about getting straight As, and math science, all you had to do was understand the stuff and you'd get a good grade, whereas, the other stuff with the subjective grading made me nervous. I was completely open about what kind of career I would have. Just I wanted it to be reasonably interesting. There's the stereotype of an engineer that as a young kid, they put together spare parts and made a computer, or built a ham radio or whatever that is. I never took anything apart.

I literally went to college without knowing how to change a light bulb when my light bulb ... Something that I would tell my father, "My light bulb burned out," and a few days later, it would be fixed, and I wasn't even curious what he had done. I assumed it was dangerous or something.

Radia Perlman:
Then in college at MIT, my light bulb burned out, and I was wandering around the dorm asking people, "Are you majoring in electrical engineering?" And finally, [inaudible 00:02:59], "Yes. Why?" And I said, "My light bulb burned out. What do I do?" So I was always very good at logic problems, but never [inaudible 00:03:08] on things. So that's sort of an important lesson that it's very important to have a team that looks at problems from different dimensions. I sort of think about the conceptual part of it. That's sort of another aspect that I have a terrible memory. So if you ask me to remember one more thing, I'm going to have to forget something important like my name. Other people manage to get through school by remembering what the teacher wanted them to say on a test, and I couldn't do that. I had to deeply understand it and be able to derive it from the two things I could remember.

Radia Perlman:
At any rate, I was good at any sort of science or math thing. Actually, I wasn't even happy about sort of being the best student. I always had a fantasy that some boy would do better than me at some math or science thing. And my plan was to fall in love with him and marry him.

Jessica Bell:
Did you ever find one?

Radia Perlman:
Well, not until college, really. And then also, I've realized that smartness has so many different dimensions, some of [inaudible 00:04:27] mutually exclusive. So you can't really compare two people. In high school, a teacher noticed that there was a programming class at a local university, so she could sign a few of us up, drive us over there, wait for the class to be over, completely volunteer on her part. I mean, teachers are just awesome.

Jessica Bell:
Wow.

Radia Perlman:
I walk into this class, after always before knowing, oh, it's a class. I know how to do classes. I'll do fine. And everyone was talking about how they had built a ham radio when they were seven, and I had no idea what that was. And then they were asking questions with fancy words like

input. I had no idea what that was. But my mind sort of closed off, and I got nothing out of that class [inaudible 00:05:17] sort of thought I was so far behind. But later on, my grade didn't matter in that class, so it was fine. But then I sort of knew that if you asked me what I wanted to do as a career, I would say, "Well, anything, as long as it doesn't involve computers."

Radia Perlman:
Then I was taking a physics class in college. And the TA said, "Would you like to be my programmer? I have a project, and I need a programmer." And I said, "I don't know how to program." And he said, "Yes, I know. That's why I'm asking you, because I have no money to pay you. And if you knew how to program, you'd expect to get paid." So I had a friend at the time who knew how to program, and I figured that was a safe way to learn. So indeed, I did learn how to program. So yeah, I can do that, but still when my daughter was so proud of me when she was in high school. Hopefully, she still is. But she was telling her friends, "My mommy's a famous computer scientist. She knows all about computers." And her friends would call me up and say, "I'm trying to install a printer and I'm getting this error message." Have to tell her that, really, there's nothing that I know how to do that any of her friends would be interested in.

Radia Perlman:
So at any rate, then I went to college. I was in math. And like you would expect, I went to grad school right after undergraduate. At the time, the MIT math department was extremely unhelpful. You had to find your own advisor. And I [inaudible 00:06:59] and insecure, and I would try knocking on a few professor's doors and say, after I'd finished all of my classwork and tests and all that, the only thing left was the thesis. But everyone whose door I knocked on saying, "I need an advisor," they would say, "Well, I'm a big, important, busy person," which was very discouraging. So I [inaudible 00:07:21] for very long before an old friend said, "Are you enjoying grad school?" And I said, "No, I have no idea how to get started on a thesis."

Radia Perlman:
And he said, "Oh, come join our group," which was a little company called BBN. And it was designing network protocols for the Packet radio net, which I discovered I loved because the concept of having all these little things all doing their own stuff independently, based on local information and how it fits together like a giant symphony is just beautiful. Oh, and then for totally bizarre reasons, I wound up in the ideal job at that point in history, which was at Digital Equipment Corporation, being [inaudible 00:08:13] of designing what's called layer three of networking, which is the thing where it enables you to plug a network together like Tinker Toys, any topology you want, and then the little connectors, let's call them switches, amongst themselves, and figure out how to create the tables that they use in order to know which direction to send messages based on the destination address.

Jessica Bell:
I realize that computing has such different implications. And we have this very small idea about who is a computer scientist and what programming is. So I find stories like yours really interesting about how you're more into the logic, or the big picture, or how the networks and the switches come together. I have the same experience as you. I took some computer science class, never had built a gaming computer and thought, "Oh, wow. I could never do this," and now I'm a software engineer. I think we have a little problem of PR in our field a little bit.

Jessica Bell:
But I'd like to continue a little bit on the idea of these networks that you're starting to talk about and move into talking about some of the things we've chatted in our intro chats about network resilience and how you transitioned from working in the networks, being in academia, to somewhere, where you are now, where you're thinking about the real world implications of how resilient a network is. What does that mean for a network to be resilient? And how do you think your background in developing STP has helped you think about networks from a higher level? Could you speak a little bit more about that?

Radia Perlman:
Yeah. Let me talk about it more historically. When I first was in charge of doing layer three for Decknet, I looked at the only existing equivalent network, which was for the ARPANET. And so I was looking at that algorithm, and I realized it's not stable. Just a few bad messages, and the network would be down forever. Now we're used to that with our PCs. It gets into bad states, and you shrug your shoulders and you reboot it. But you can't reboot the internet, especially the way that you diagnose what's going on, and the way you fix it is by sending network management messages [inaudible 00:10:45] network.

Radia Perlman:
So my first paper was about, hey, it's not good for a network to be so fragile that a few corrected messages makes the network be down forever, and this is how to fix the algorithm so that it would be self stabilizing, meaning that we get rid of the machine that is generating bad messages, the network will recover by itself, whereas the ARPANET algorithm would not do that. And indeed, actually, it did wind up having that problem. There were a few corrected messages, and the network was down forever. And the only way that they managed to fix it was that they were extremely lucky. The same people who designed the algorithm and wrote the code were the ones fixing the network, so were able to do a core dump, realized what the problem was, and basically have to come up with a patched version of code that specifically ignored the kind of messages that were corrected, and then they could one by one, get rid of the

bad messages that were floating around. But you couldn't do that in the internet today, so it's very [inaudible 00:12:05] that it be self stabilizing.

Radia Perlman:
Another dimension was that original networks were for researchers to share data. And so it was fine for the people managing the network to ... That was their entire job, and they designed the algorithms, they wrote the code. They understood it all. But these days, and shortly after that point in history, real people have to keep the network working, so it has to be easy to manage and intuitive to manage. And if you have to set parameters, it has to be very resilient to mis-configuration. So that was kind of another dimension that I added. When I designed things, I was designing it for somebody like me, meaning I don't want to even think about it. You just want to plug it together and it will work.

Radia Perlman:
Somebody once complained to me, once told me that they had customers that were complaining about one of the products I had designed. And they said they find it so boring because all you do is you plug it in, and you don't have to do anything with it. Another thing that once someone told me was they were doing a survey of customers on how much they liked Decknet, and most of them answered, "What's Decknet?" Which actually was a great compliment.

Jessica Bell:
Right. Right.

Radia Perlman:
They don't even know they're using it. So another dimension of resilience was in that original paper, that last sentence I had said was, "This is how to design a network," so once whoever is injecting bad messages gets disconnected from the network, the network will return. But there's no way to make a network continue working while bad things are happening. And for weird reasons, 10 years after I dropped out, I went back to grad school. And my manager at Digital, who was one of the smartest people I've ever worked with, suggested as a thesis topic, that I either prove that statement that I made in the original paper, that it was impossible to make a network work, or that I design a network that will work even if some of the components are malicious and are [inaudible 00:14:38] generating as bad messages as possible, so that indeed was my thesis.

Radia Perlman:
I've never done anything terribly complicated. Everyone thought it was going to be a hard problem and an important problem, so they told me certainly that would be good enough for a

thesis, even if you can only do a very small subset, like a lab of two computers that are direct neighbors of each other to be able to communicate, even if someone way far away is malicious. But it turned out to be very simple. And then I was nervous that the solution was so clear and simple that they wouldn't give me a thesis for it. So I asked somebody, "Is there any minimum length to a thesis?" And he told me, "Well, it either has to be long or good." So there's a lot of dimensions of resilience that I care about. So one is usability that a lot of times, things don't work because the users did the wrong thing. And I claim you should never blame the users. You should design your things so that there's nothing the user can do to break it.

Radia Perlman:
Then there's the just plain self stabilizing, but then there's the working despite having some components being malicious. With the internet, it's just too large to assume that everybody has good in their heart.

Jessica Bell:
Right. And especially nowadays with our networks going beyond what we ever thought at the beginning. I mean, these networks are now spanning every single part of our world and handling data that we never thought. It's not academic anymore. Right? It's everything, credit card information, public identifiable information, all this kind of stuff.

Radia Perlman:
Early papers on network security said what you have to worry about is theft of data, unauthorized access to data, unauthorized modification of data, and oh, yeah, there's denial of service, but we don't have to worry about that because no one would bother because you don't gain anything from doing that.

Jessica Bell:
Right, right. Which is [inaudible 00:17:05] now.

Radia Perlman:
Right. It's proven to be absolutely false. You actually can make a lot of money from denial of service.

Jessica Bell:
Right. What do you think drew you towards thinking about network resilience, of all the things you could've done with computer science, or your early career? What kept drawing you to this idea of networks and their overall resilience?

Radia Perlman:

I think it was just where I landed. If a different friend had approached me when I was I and grad school and said, "Are you happy in grad school?" "Oh, really? You're not happy now. Why don't you come join our group?" And it was chemical engineering, I suspect I would be just as happy in chemical engineering.

Jessica Bell:
Right. Right. So how has this idea then shaped ... Once you got into this and once you started realizing, oh, I actually ... This is simple to me. I have these really clear, simple ideas, which I fight against this all the time. I'm always like, "I want more simple code, not more complex code." But that's a whole nother conversation. How do you think that this has now gone on to shape some of your further work, and the work that you're doing now? Have you always stayed within this vein? Or did you end up going along different sort of branches of a tree, if you will?

Radia Perlman:
I still wind up when I design something, that after I design it, people will look at it and say, "Oh, yeah. That's very simple. Anyone can do that." And I sort of know from experience that even if I do something incredibly simple, that if somebody else had designed it, it would most likely be an extremely complicated thing that wouldn't work in all cases. And whenever they discover a new case that it doesn't work in, they add an extra little piece to it to try to handle that case. It's I think partly because I get rid of all of the details that are irrelevant to get to the heart of the problem being solved. Or if it's a big problem, I will partition it into smaller problems and then solve each one simply.

Jessica Bell:
Yeah. Who was that designer that said, "Good design is when you take everything away. And when you can't take anything else away, you've got a good design"? Oh, man. What is his name? He's a German guy. He's very famous. Anyways, it sounds like that.

Radia Perlman:
[crosstalk 00:19:41] or something, but yeah.

Jessica Bell:
It sounds a lot like that.

Radia Perlman:
It might've been Ben Stein that said, "Keep things as simple as possible, but no simpler."

Jessica Bell:
Yes, exactly. Exactly.

Radia Perlman:
What we talked about had to do with the spanning tree algorithm. It's sort of ironic that, that is what I'm mostly famous for. But it has a nice ring, spanning tree algorithm, whereas resilience and scalability and manageability of network protocols is really kind of too long of a phrase. It's not very catchy. But the spanning tree algorithm was a hack that I thought was a bad idea. What happened was that in the network, in order to send a message, you have to put it in an envelope that says who you want it to go to. But then people came out with the ethernet with fanfare, and everyone was so excited about this new way of doing networking that they were doing that instead of putting in the kind of envelope that enables it to get forwarded across a large network. Ethernet was really just a single wire that everyone on the wire could hear everybody else. So there was none of this forwarding necessary.

Radia Perlman:
And they were doing that, I was saying, "No, no, no. You still need layer three." And they would say, "Oh, Radia. You're just upset because no one needs your stuff anymore." I said, "But you may want to talk from one ethernet to another." And they said, "Oh, our customers would never want to do that."

Jessica Bell:
Of course not.

Radia Perlman:
So years later, when, hey, our customers did want to talk outside the scope of a single building, my manager said, "Hey, Radia, we need to design a magic box that will sit between two ethernets and forward packets from one to the other." Of course, what my stuff did, but the stuff I did required the end notes being aware of it and using that envelope and doing other things. So the constraint was design something that does not require changing the existing end notes, and does not change the ethernet header in any way. There's no spare fields, and there's a hard size limit, so you can't add anything there. And so that's kind of where I came up with the spanning tree algorithm, which took exactly one week.

Radia Perlman:
Now to not only invent it, it's a really simple thing, but to write the specification, my manager asked me to do this on a Friday and then disappeared for a week. He was on vacation. And this was before you could send email or call anyone on cell phones. So at the end of Tuesday, I had the complete spec written. And the implementers got it working very quickly from that spec without asking me a single question. So the remainder of the week was used on writing the poem that goes along with the algorithm. And it was abstract of the paper in which I published it.

Radia Perlman:
Now I still thought that the right answer was fixing the end notes to put layer three in. But it was fine to do this thing because it might take six months to do that. But it's interesting that even though now everyone has put layer three back into the network stacks, everyone is still using spanning tree. And as a matter of fact, ethernet, the original invention, doesn't exist anymore. When you talk about ethernet, it has nothing to do with the mechanism for sharing bandwidth on a single wire, which was the original invention. It's all just point to point links, just two notes on a link, and forwarding using the spanning tree.

Jessica Bell:
Wow. I mean, it's remarkable to think about these things that someone like me learned of as just the foundation of the internet. And to hear that you made it over a long weekend is just mind blowing to me.

Radia Perlman:
Well, it isn't like every week I come up with something that changes the world.

Jessica Bell:
That's true. That's true. It seems like sometimes the most impactful pieces of technology that we build are not the big, super complicated ones. It's the small, simple, connecting two things, or filling this gap that we think it kind of silly or frivolous. And then all of a sudden, this changes everything in the whole field. I feel like the more I learn about computer science history, the more I find most problems are quite simple. It just takes the knowledge of, well, we should hook these two things up, or we should fill this gap here.

Radia Perlman:
Yeah. Complicated things will never really work. I was once at a meeting where somebody was talking about something that was just so bizarrely complicated. And [inaudible 00:25:14] was sort of silent, and kind of depressed about this thing. And somebody said very cheerfully, "Well, with enough thrust, anything can fly."

Jessica Bell:
That's true I guess. It's just: How long will it fly before it crashes to the ground?

Radia Perlman:
Exactly. Yeah.

Jessica Bell:

So I'd like to sort of use this idea to transition a little bit to something we've talked about as well, which is computer science curriculum and the industry, and sort of how we're potentially missing out on certain things because of perhaps it's a PR thing. Perhaps people stay away because they think, "Oh, I'm not tinkering with a ham radio." Or these days, I'm not building my own video game in high school. That people sort of shy away from the field, or the idea that perhaps our field is too focused on certain things. I know you have a lot of thoughts around these. So I'd like to hear a little bit about: What do you think is missing from our curriculum, or perhaps our industry as a whole?

Radia Perlman:
Well, something that I'm passionate about is critical thinking, so surprisingly, not everything you read on the internet is true. But it's not just everything you read on the internet. It's even textbooks, wrong, and your professor, or whatever. Well, the early education that I had, everything was true, false, multiple choice, right, wrong. And it's kind of a very bad way to teach thinking because in general, problems are not like that. So getting people to look at something and say, "Well, in what ways can you imagine this being correct? How can you argue for this? How can you argue against it?" And make them think on both sides. Have them read papers that were published in reasonable venues that turn out not to be correct. So they should just be questioning things as opposed to thinking that a teacher's job is to fill your head with facts, and then test that you are remembering the same facts that were fed to you.

Jessica Bell:
And how do you think that might express within our field? Because I think computer scientists might have an image of themselves that, oh, well, we are very critical thinking. Right? We have to think through these big, critical ideas. And yet, perhaps that isn't necessarily the case. How do you think a wider view of critical thinking could be introduced into our field, either through education or through things like bringing in people who necessarily wouldn't come to the field, or haven't been trained in a certain way? What are your thoughts around the subject?

Radia Perlman:
Yes. So a lot of people that I've talked to that say they're going to go into French literature, I say, "Well, why not STEM?" And they say, "Well, I wouldn't be good at that," or something. And they don't ... One of the reasons I'm so valuable is that I have found an ecological niche where I'm very different than the other people. And I look at problems kind of a different way, so that's useful. The problem is you need to have a corporate culture that lets people thrive. That's another whole passion that I've had. And I've been observing how organizations work and don't work, just like I look at computer networks and think of how to make them [inaudible 00:29:10] or whatever.

Radia Perlman:
So one rule is that it has to be safe to ask questions. But an example of a corporate culture where it was exactly the opposite was that I was in a group where the culture was dominated by some very aggressive, obnoxious people. I have yet to see anyone that acts that way, by the way, that's actually good technically. But at any rate, the culture in this group, if you were to ask a question, one of those people would say, "If you don't know that already, you don't belong in this group." Which, how do you go from there? So if somebody asks me a question that everyone knows, like: What's a public key? I won't say, "How can you not know that?" I'll say, "Oh, my goodness. It is the coolest thing. And I can't believe that I have the honor to be the first person to introduce you to it."

Radia Perlman:
But another dimension to that is if you're a senior person, sometimes you think, "Wow, I'm supposed to know everything." And if you believe you know everything, you're incredibly dangerous, and you should retire. But if we're a really senior person, you should be a role model, and you should be the first person to ask naïve questions. I don't know what that is, to show that you're perfectly comfortable admitting that you don't know everything, and that it's fine to ask questions. If you bring people in who have not been living in your little specialty for their entire lives. Sometimes they'll ask questions that make you rethink your basic assumptions, [inaudible 00:30:56] basic assumptions are wrong.

Jessica Bell:
Right. Exactly.

Radia Perlman:
So it's very, very useful to bring people who are intellectually curious, don't [inaudible 00:31:09] jargon. So how do you manage to explain the conceptual part of what you're doing without jargon?

Jessica Bell:
Right.

Radia Perlman:
And merely that exercise of explaining it at a conceptual level is useful to the person explaining it. But then answering questions from people who you've intrigued, and they said, "Well, couldn't it have been done this other way?" Right, is just a great opportunity for creativity and innovation.

Jessica Bell:

Definitely. I feel like that has been so true during my career. I have a degree in international relations and politics, so I don't come from software at all. It's always been the people who sit down with me and explain things to me that I have found to be the most curious and effective programmers because they're not afraid when I ask the question, "Well, why don't you do it this way?" They're no afraid to say, "I don't know why I didn't do it that way. Let's think through that process." And it kind of comes back to your point about resilience. You'll create more resilient things if you allow all of these questions and all of these different thoughts to come in and don't close yourself off. Your software, or your product, or your network is going to be a lot more susceptible to staying up versus down, or something like that.

Jessica Bell:
I love your take on, I'm so honored to be the first person to share this with you. That is such a wonderful take of being a mentor and somebody who gets people excited about our field, which can be challenging.

Radia Perlman:
Right. People always talk about mentoring. I'm always a little bit uncomfortable with that. Of course, I mentor people all the time, but only about things that I think I can help them with. So if you want to know how to write a book, or to look over your design, or something like that, yeah, sure, I can do that. But if you want to know which car to buy, I know wheels are important for some reason, not much else. So I think everyone should be mentoring everybody at all times, rather than the notion of kind of a linear thing that I am more senior than you, so therefore, I am the mentor and you are the mentee.

Jessica Bell:
Exactly. It's that back and forth relationship. The mentor is getting as much as the mentee, as the mentee is getting from the mentor. You're sort of together, forging ahead in this area that you thought and you do know a lot about. It's more of a relationship than a linear thing, like you said.

Radia Perlman:
Right. And actually, the most powerful way you can grow the next generation is asking them for help. So rather than you graciously sharing your wisdom, the good thing to do, but it's also letting them know that they have wisdom that you can learn from as well.

Jessica Bell:
Exactly. Exactly. That's a great segue into our final byte question, which is looking towards the future. And I'd like to hear from you. What are you most excited about, scared of, or genuinely

curious about in our field over the next say, five-ish years? What does the future of computing mean to you? Yeah, I'd love to hear your thoughts.

Radia Perlman:
Well, if you asked me about five years ago about the internet, I would just be so spectacularly excited about how it had transformed society in so many ways I couldn't even have imagined, that all information is available, that a merchant can reach a global audience without even needing to create a storefront. And you can [inaudible 00:35:15] that you can only find in some obscure part of the world. This was all so magnificent, and keeping in touch with friends and family, and all this was wonderful. That was five years ago.

Radia Perlman:
Now I actually think it will be the end of civilization, or maybe it already is [inaudible 00:35:34]. You no longer know what's true anymore. Back in the days when reporters were the camera crew from channel five news, you kind of knew who they were. They had credentials. But these days, everybody is a reporter, anyone that has a cell phone. But also, you can actually create false videos and false information. That's happening all over the world, so it's important that any citizen can let the world know what's going on. But you don't know whether that information is true. You can't do it sort of cryptographically by proving who the source was because the source was some unknown person. So the end of truth is kind of scary.

Radia Perlman:
The fact that you can virally spread horrible mis-truths on purpose is [inaudible 00:36:44]. The fact that there's all these AI bots going around trying to figure out what keeps you interested. If there's an article that you seem interested in, they'll give you more and more things like that, and more and more extreme examples. And that's what so terrifying about how it's polarizing society to an extremely dangerous extent. So I don't know how to get beyond this, even simple security things. The theory of security is just so beautiful, where [inaudible 00:37:25] to my bank. It has a DNS name. And there's a certificate and public keys and wonderful protocols, wonderful map. All this works great in theory. But in practice, I actually fell for a scam recently, which made everything so clear to me that, oh my goodness, none of this stuff matters, which is I wanted to renew my license.

Radia Perlman:
And I knew you [inaudible 00:37:53] online, so I just put into Google, renew Washington state driver's license. And I clicked on the top result, and it asked, looked perfectly reasonable. If I had bothered looking at the URL, the DNS name was something perfectly reasonable like Washington.org. And so I had a choice of renew license, replace license. And I clicked on renew, and it asked me all the questions that you would expect, my address, my name, my

license number and my credit card. Here's a bunch of offers you're qualified for, which was my first hint that it wasn't the right site. As a matter of a fact, they'd been better off just saying, "Expect your license in six weeks." Of course, they never gave me a license. If I'd read the website more carefully, it just said it would give me instructions for how to get a license.

Jessica Bell:
Oh, goodness. [crosstalk 00:38:54].

Radia Perlman:
One of the offers was a free eBook I could download that would tell me how to get a license, which presumably would've said, if they were at all honest crooks, would've said, "This is the URL you should've gone to."

Jessica Bell:
Wow.

Radia Perlman:
There are similar websites like that. And how is a human supposed to know the right ... Do you expect a human to look at this three line URL with all these weird characters and figure out which part is the DNS name? We shouldn't be asked to do that. So it's basically security in so many different dimensions that it's terrifying and absolutely must be solved in order to ... So that means the good news is for students, there's so many dimensions in which they can be useful. But the bad news is that it really is very scary.

Jessica Bell:
Right. Right. Well, that's a really honest answer. And I appreciate that. I appreciate the honesty of the people who are sort of watching over the whole field because I have the same fears and anxieties. Well, Radia, this has been so wonderful to talk to you. I've really enjoyed having you here. I wish you good luck in your further career. And thank you so much for writing such a great peek into your mind and your thoughts, and where our whole industry has come from. I really appreciate it.

Radia Perlman:
And thank you so much.

Jessica Bell:
ACM Bytecast is a production of the Association for Computing Machinery's practitioners' board. To learn more about ACM and its activities, visit acm.org. For more information about this and

other episodes, please visit our website at learning.acm.org/bytecast. That's learning.acm.org/bytecast.