Scott Hanselman (00:01):

This is ACM Bytecast, a podcast series from the Association for Computing Machinery, the world's largest education and scientific computing society. We talk to researchers, practitioners, and innovators who are 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 today, Scott Hanselman.

Hi, I'm Scott Hanselman. This is another episode of Hanselminutes in association with the ACM ByteCast. Today I have the distinct pleasure of chatting with Eric Allman. He is a computer programmer who developed sendmail and its precursor delivermail in the late 1970s at UC Berkeley. And in 1998, he and Greg Olson co-founded Sendmail, Inc. How are you, sir?

Eric Allman (00:50):

I'm doing pretty well. It's a lovely day.

Scott Hanselman (00:52):

It is a lovely day. I woke up and I thought to myself, I'm going to try to get some sun and some vitamin D today. And I think that as programmers, I'm sure it's a constant balance between working on the thing that captures your attention and trying to get some sunshine and touching grass.

Eric Allman (01:07):

Yeah, I have to admit, I'm looking out my window at the garden right now.

Scott Hanselman (01:11):

So you are currently retired. I'm curious what made you retire? Some people just work until they finally fall into a hole.

Eric Allman (01:20):

I think I would probably have to say it had a lot to do with—actually, a lot of things. I retired basically during COVID. I was working at UC Berkeley for my third job at UC Berkeley. Third and presumably last. And the last grad student on the project I was working on graduated. And the new grad students didn't show up because of COVID. But they were working together, and I had worked on a project with this group, and I had assumed they were going to leverage off the code that I had already written—I and the other people in the group—and they didn't. They wanted to go in a totally different direction. So at some point I went, they're paying me to do nothing. Or nothing of use. I mean I'm available, but that didn't sit well with me. And I was at retirement age, and my husband has a degenerative neural disease, so he is slowly losing the ability to walk and he wanted to do traveling as much as he could while he still could. And so that seemed like another good reason to do it. And a bunch of things just came together all at once.

Scott Hanselman (02:32):

And have you been able to do that traveling?

Eric Allman (02:34):

Yeah, actually we've done a couple of trips. Several trips actually. And we've got two other big ones coming up this year including a train tour around Africa.

Scott Hanselman (02:46):

Oh wow, that's amazing.

Eric Allman (02:49):

Yeah, should be kind of fun.

Scott Hanselman (02:51):

So I've been married for 25 years. You're coming up on 13 this year, is that right sir?

Eric Allman (02:57):

Actually, based on the beginning of our relationship, it's more like 44, 46, somewhere in there.

Scott Hanselman (03:04):

Okay, that's amazing.

Eric Allman (03:05):

We got together in ‘79.

Scott Hanselman (03:07):

Oh wow. That's fantastic. And your husband Marshall is a well-known computer scientist as well. And I'm curious because my wife and I—she's nontechnical, she's a nurse. And we were writing a book on what it's like to be in a mixed marriage and the joke is the mixed marriage is I'm technical and she's not. Is two technical people in a relationship challenging? Do you chat about computer stuff or do you just keep that part of the relationship separate?

Eric Allman (03:32):

No, I think you could safely say that we chat about it. I mean, he’s got his stuff that he does. He's basically a file systems guy. I don't do file systems, but, you know, he doesn't do email, so. Actually I don't do much email anymore other than getting far too much of it and so forth. I use it, but I'm out of the development cycle now and in fact, my main gig is a 501c3 that I helped found called the Berkeley Historic Building Fund, and we raised funds for preservation of historic buildings starting with the Hillside Club, which is just up the street from me.

Scott Hanselman (04:14):

That's fantastic. Is that amazing though, to let go of so much computing? I'm just really, as I start thinking about when I'm going to retire, do you just release it into the universe and you just say, I'm just not up on the latest stuff, or did you cling to trying to stay current and stay up to date?

Eric Allman (04:32):

Well, I've got an ability to concentrate, I guess you'd say. I've done many, many things. So various stuff I've done—I started off really doing database systems and was the lead programmer on the Ingress Project at UC Berkeley. That was my first university job. And I've done sort of user interfaces to database

systems, which I think I overreached for the ability of the technology to support it at the time. So that one never got out as a product. And I worked on neural networks at International Computer Science Institute for a while. So that's basically the same technology behind all the AI you see today. But back then a mega flop was a really, really, really fast machine and now it's like they're giga flops. So that one, we got some interesting papers out of it. It was an academic organization, but the goal was continuous speaker, independent speech recognition, and we never really got that to the point where it worked to our satisfaction. Now of course my phone can do it.

Scott Hanselman (05:40):

What does that feel like? Because I feel like I can see so much history—and I've only been doing this 35 years—to be sitting at the tail end of this and go, wow, we really built a deep stack of stuff. We're teeter tottering at the top of this pyramid of all the different protocols that came before us and each additional layer of abstraction is indistinguishable from magic. And now I've got this pocket supercomputer that can recognize my voice and do stuff.

Eric Allman (06:10):

Well, I actually started on the ARPANET and that was a very different network. They only felt they needed 8-bit host addresses.

Scott Hanselman (06:20):

Should be enough for everyone.

Eric Allman (06:22):

Yeah, exactly. And what was it, Thomas Watson I think, said the international market for computers is going to be about 100.

Scott Hanselman (06:31):

Be a mainframe on each country. And that would be it. Each country would have their own mainframe and maybe one per continent. Should be fine for everyone.

Eric Allman (06:38):

Yeah, so I've seen a lot of stuff happen. It's been kind of interesting. When the network first appeared, we, I think, had this fantasy that we were going to bring understanding to the world. When everyone could communicate with everyone else, then wars would be gone and so forth and so on. We really had a concept that this was going to be a utopian future. Now that it's the future, I don't think it looks quite as utopian as we used to think it would be.

Scott Hanselman (07:11):

That's disappointing. It does feel like eternal September really is kind of eternal at this point. There was a smaller kind of feeling of community and neighborhood in the early days. When I first got online in the late eighties, mid eighties, late eighties, it just felt like there was maybe a couple of thousand of us and we were all pretty pleasant. And now it seems somewhat chaotic.

Eric Allman (07:33):

I actually have a book in the other room, which was essentially a phone book for the ARPANET slash early days of the internet with the name, address and phone number of every user on the network. And I'm in it.

Scott Hanselman (07:48):

Yeah, I had a Fido net node for a time that I ran in my bulletin board and was, you know, making local phone calls trying to not pay long distance phone calls and bouncing email rather around or Fido net mail at least for a time. I had a spiral notebook with kind of the 30 or 40 folks that I knew around. It felt like Ham radio, you know? It was very small.

Eric Allman (08:11):

Yeah, I guess Ham radio still is.

Scott Hanselman (08:14):

Yeah, I guess. Well because it's hard. You got to get a license. Which is an interesting idea. In fact, maybe they should have given us a license to send email.

Eric Allman (08:22):

That would've solved a lot of problems.

Scott Hanselman (08:25):

Yeah, indeed. Were you thinking planetary scale when you were building these systems, when you and your compatriots were doing things like sending mail? I mean email is kind of the original open adversarial system. It's open, it's federated, it's effectively unauthenticated—or at least it was at the time—but it's also constantly attacked. And I'm curious, were you thinking this is for everyone for the world right now or you were just building it for you and your friends?

Eric Allman (08:51):

So there was already a network known as UUCP and that was dial-up modems. And it turns out that I know the guy who first brought UUCP to Japan. I know the guy who brought the first UUCP to Europe. And Australia actually.

Scott Hanselman (09:11):

Is that Mike Lesk?

Eric Allman (09:12):

No. Teus Hagen in the Netherlands, Robert Elz in Australia. The point is we were very early on thinking there's no particular reason why this can't be an international thing, which I think part of the reason we really thought it was going to make the world a smaller place and everyone would discover that we're just people and so forth. Like I said, fantasies.

Scott Hanselman (09:38):

Sendmail is somewhat famous for being extremely configurable. Maybe a little complexity crept in there over time. Do you see any parallels between sendmail and sending mail and the kind of retries and

queues and storing forward, and then what we're seeing in agents and things like that in AI? Do you think at all about how we're all just building the same thing over again in a different font?

Eric Allman (10:01):

Actually, I hadn't thought about the comparison to AI. I probably should have because the reason sendmail is so configurable is because it started before the internet. I was trying to get the local network at Berkeley—which was called, creatively enough, Berknet—and UUCP and the ARPANET and so forth to talk to each other. And they all had different address syntaxes and different semantics and so forth. So sendmail was designed to be the glue that would hold them together. And I had taken an AI course and one of the things we talked about was production systems. And I said, I know AI is maybe overkill for an email program, but production system looks like it fits really very nicely. And in fact, that's the core of the sendmail configuration file. So it's been proven that it's Turing complete, which probably doesn't need to happen. And the truth is, I mean the world pretty much agrees now that everything should be user at sign domain. Nobody uses host bang host bang user or host colon colon user or any of those others. So yeah, it was basically setting up something where you could use effectively an AI technique to route email. But it hadn't occurred to me anything comparing it to large language models.

Scott Hanselman (11:38):

What do you think is a design decision in sendmail that may have aged better than you expected? You don't really know if these things are going to be extensible, if they're going to hit some kind of a design wall. But looking back, is there a particular design decision that maybe aged quite well?

Eric Allman (11:54):

I mean there was a couple of things that I did kind of on the side not as well known as my background with sendmail is that I also did the first implementation of Syslog and I did that as a tool to use for delivermail, the predecessor to sendmail. And at the time we didn't have Interprocess communication and so forth, so I had to use things called MPX files and then IPC came along and I converted it to use the new technology and so forth. That's proven to have real legs. I mean there's dozens of implementations of Syslog all basically at their core, the same thing that I did as part of the implementing sendmail.

Scott Hanselman (12:42):

Yeah, that's a great point. I mean Syslog, Syslog D, these are something that is keeping track of everything for the last 50 years, 60 years. And now on top of that we have new ideas like open telemetry and I'm seeing people shuttle their logs around with things like that. That seems like a pretty successful project.

Eric Allman (13:06):

Well, it wasn't a project in and of itself, it was really just part of a much larger project. I'm just thinking if there's anything else, to be perfectly honest, the sendmail rewriting rules, which are the things that people complain about the most because they're big and complicated—I totally understand them, but that was very successful because it made sendmail be the glue that could hold the universe together as it was still in this nebulous state and still an astral cloud as opposed to a solid planet. So I think the uptake of email would've been much slower if people had been stuck with sitting in these pools that didn't talk to each other.

Scott Hanselman (13:52):

And the fact that sendmail rules are recursively applied, I think some people have found that to be somewhat mysterious and frustrating, but that's kind of the genius of it.

Eric Allman (14:02):

I mean, you can use recursion, but it's not necessarily necessary in order to do the job.

Scott Hanselman (14:13):

But that level of flexibility was kind of the secret sauce.

Eric Allman (14:17):

Yeah, it was also part of the thing that made it sometimes really challenging to debug.

Scott Hanselman (14:26):

ACM ByteCast is available on Apple Podcast, Google Podcast, Podbean, Spotify, Stitcher, and TuneIn. If you're enjoying this episode, please do subscribe and leave us a review on your favorite platform.

You said that you were doing AI. Like, AI had a moment, and I think AI has had a branding moment, but you were thinking about AI and your compatriots and your peers in the space were thinking about AI a very, very long time ago. Did you all, in the creating and thinking about neural networks, see transformers and generative pre-trained transformers coming? Was this an inevitable thing that would hit the hockey stick and then take off? Or did you kind of wake up when ChatGPT happened and you said, oh crap, they did it?

Eric Allman (15:15):

More the latter. When we were doing it, we were just trying to get it to run fast enough to implement a relatively small neural network, and we weren't using AI in general. I don't think we even ever called it AI. It was just neural networks, a name that seems to have gone out of favor, but is still there at the core. The new models are so much more sophisticated. They've got callouts to engines that are not stochastic parrots and so forth. So I've actually started playing with Claude to do some coding stuff. I do full code some at home and it's really stunning. The thing that really became clear to me is I can only absorb so much stuff in my head and asking Claude to do these things, it said, oh yes, there's already a library to do that. Here's how to use it. And it's like, I've never heard of that library before and you're right, it does exactly what I need. So that alone has me as a bit of a fanboy these days.

Scott Hanselman (16:22):

It is extraordinary. I feel like Opus last October kind of was the moment and Claude Opus is an extraordinary model and I kind of go back and forth and I get a little bit of an analysis paralysis because I think to myself, maybe I'm a stochastic parrot and maybe what I'm about to say to you is the statistically most likely next token based on the last 50 years of my life and the context window that I carry around with me. Maybe I'm just saying the next obvious thing. So then you get all in your head about are they parrots and they're just saying code because they've seen code or is there more there? I like the stochastic parrot like analogy. I think it is a good one because I refuse to believe that it is in any way conscious or creative.

Eric Allman (17:07):

Here we start getting into the realm of philosophy as opposed to computer science. What is consciousness anyway?

Scott Hanselman (17:13):

I think you're qualified to have an opinion, though.

Eric Allman (17:15):

I'm qualified to have an opinion, but it's just my opinion and hence it's worth the price you paid for.

Scott Hanselman (17:21):

I'll take it though.

Eric Allman (17:23):

So it certainly presents as though it's got some kind of consciousness. It carries on a discussion with you and so forth. But for example, I've had cases where Claude will say, oh, to fix that bug, you do this and then there's something else. And Claude fixes that, but puts the other bug back in. And I don't think that exhibits as consciousness. Claude is very helpful as long as you use it effectively, but I can't see doing vibe coding where you just say, here, throw me some code. I'll test it and tell you if it works or not and if it doesn't, try again.

Scott Hanselman (18:09):

It is so funny though that we made programming languages and syntax and we worked so hard to make ourselves able to express our intent clearly and unambiguously to the computer. And now we've created the prose compiler where one can just talk in English or any language, but if there is a lack of clarity, it will fill that lack of clarity with the statistical mean or just randomness. It'll come up with something, it'll vibe it. And then we are frustrated when it doesn't work out because were we not clear or was it that it filled our lack of clarity with its randomness? And then I kind of waffle back and forth between I don't want to remember all the details and I don't want to type all this stuff. It's still fun, it's still great. It's frustrating. I waffle back and forth between love and hate for AI-assisted coding.

Eric Allman (18:59):

I can certainly understand that.

Scott Hanselman (19:01):

Do you code primarily just for projects? Do you have an idea and you think I'm going to code this or—I know you have a garden, do you like, oh, I'm going to code some IOT internet of things for the garden? I have all kinds of projects that I'm working on just to make the house run better.

Eric Allman (19:15):

Well, I think this counts as making the house run better: Kirk and I have a wine cellar, and so at some point, a long time ago I started keeping a database, which was just a flat file—parsable, but a flat file—and at some point I wanted something fancier. So I built myself a wine database with a GUI interface and so forth and so on. And periodically we wanted to tweak that. So there's that. There's some home automation things which involve code I find myself writing in C++ for the Arduino and usually some other higher level language for general coding stuff that's going to run on my server and whatnot. I use

MQQT a lot for getting data around, so things don't have to be all on one piece of hardware. You can just talk to the broker and get what you need. So Python has been probably my most common language, even more so than C, which is a little shocking. I was a dedicated C programmer for most of my career.

Scott Hanselman (20:28):

It is surprising. I'm a C# programmer, but ultimately any curly brace in a storm, honestly. I think we could probably argue about whether we should use the almond indent style or whether with the correct way to do indentation and do curly braces. So I do C#, but I keep falling back into Python. It's just kind of simple and easy. And when I do my home automation, even just as a point of note, I'm a type one diabetic and I have an insulin pump and a sensor and it all kind of runs on Python and I've got Python systems that will go and check my blood sugar and notify my wife and tell me if things are going to be okay. And certainly I could have done it in any of the curly brace C languages, but I keep falling back into Python these days.

Eric Allman (21:11):

Well, Python's added enough—basically a type system—that it's becoming…the problem with Python when it's in pure dynamic mode is you never know for sure whether your code's debugged or not, and a strong type system can really help that. So I haven't really made the switch to typed Python. I'm aware that it exists, but I haven't read up on it enough to know how to use it effectively.

Scott Hanselman (21:36):

The same thing of course happening in JavaScript with TypeScript where you kind of go back and forth between how's drawing type, there's kind a slider bar of typedness and how much you want to care as you move from JavaScript to TypeScript.

Eric Allman (21:48):

I've never been a JavaScript fan, let's just put it that way.

Scott Hanselman (21:52):

Well, so I've always felt like, I like to joke that JavaScript is not the language that we deserve, but it is the one that we have. So it's an imperfect language, much like English, but it's the one that we're all speaking unfortunately.

Eric Allman (22:06):

Yeah, well some of the automatic type conversions between things that—they are strings, but they might be numbers, so it will just Oh, it's a number now, and that's not what I always want.

Scott Hanselman (22:22):

No, it's not. So you might enjoy TypeScript. TypeScript of course came from Anders Hejlsberg, who made Delphi and C#. So he was basically finding JavaScript equally objectionable as you are finding it and said, alright, what if I applied what we know about C# and strong typedness to JavaScript and really locked it down. So I think most of the JavaScript that people are writing these days has some common sense typing, I think.

Eric Allman (22:50):

Well, thank goodness for that.

Scott Hanselman (22:52):

Thank goodness for that, indeed, sir. What have you been doing over the last five or 10 years when it comes to coding and your own learning? I'm just trying to plan for the remaining years of my career and I'm wondering, will I just become a tinkerer or will I go back to school and teach? Because I feel like I know a lot of stuff and I want to share it with folks. And I'm wondering when you were doing your work in academia, if you found that to be really satisfying?

Eric Allman (23:20):

Well, I do have to say the best part of my last job working on a research project, again, was the students. The students are just so bright and so creative and so forth. There was one of the students—I usually came in more in the morning. I'm more of a morning person I guess—and he would wander in around noon and immediately start talking with me about whatever it was he had been thinking about. And it's like half the time I didn't know what he was talking about. But it was so great being bombarded with this pretty much every day. So I haven't been seriously trying to keep up on the absolute most current version of everything, partially because I'm traveling a lot more. And that feels like too much work to be something you do on vacation.

Scott Hanselman (24:16):

It's going to be fascinating. I have a number of friends that are telling me that I'm fooling myself into saying that I'll retire and that I may fall in and out of retirement depending on what feeds my spirit and what works well for my family at the time. So it's going to be interesting to see if I can rise to the challenge.

Eric Allman (24:32):

Yeah. Well, I actually did retire several years ago, or at least I thought I had retired and at some point got bored. And actually that was right about the time I went back to the university. I got a call from somebody I had worked with in the past who I really liked working with, and he called and said, so I'm working in a new lab up here on campus and we got seminars that you might find interesting. And oh, by the way, free lunch. And so I started going to those and ended up hanging out in a research group just, they said, hang out if you want our meetings right after the seminar. And at some point they said, if you're going to be here this much, we really ought to pay you. So that was the beginning of my last job.

Scott Hanselman (25:20):

That's a pretty good gig. I mean, if you can get snacks and also be paid for it and you enjoy the work, that's kind of the optimal scenario.

Eric Allman (25:27):

It is kind of, yes.

Scott Hanselman (25:31):

Well, I admire you very much. I think that I hope you appreciate how much people in the circles that we run in appreciate you. Of course. You're an ACM fellow, you've done a huge amount of work. Your sister is also well known, has been doing a lot of open source software since the eighties. I hope that you give

her our best and also to your husband, Kirk. We appreciate you all and the work that you've offered us over so many years.

Eric Allman (25:56):

Thank you. I do find it amusing that my husband, myself, and my sister all have Wikipedia pages.

Scott Hanselman (26:04):

That's kind of a pretty cool flex. I mean, you have to admit at Thanksgiving dinner the number of Wikipedia pages is probably three for a long table. Is there anyone else we should know about with a Wikipedia page in the family?

Eric Allman (26:17):

Not that I'm aware of.

Scott Hanselman (26:19):

Well, they need to get on your level and work a little harder than, don't they?

Eric Allman (26:23):

Yeah. Well, I have to admit, I feel like I've played most of my career and not really worked much.

Scott Hanselman (26:29):

And that is the goal. And if you can say that with a straight face, then you have lived a blessed life, sir.

Eric Allman (26:36):

Yeah, except for email standards. I was doing standards for a while and decided I really, really did not like doing standards.

Scott Hanselman (26:45):

The standards committees are a lift. I will admit that is a challenge.

Eric Allman (26:50):

So I retired from that, but not from my job at the time.

Scott Hanselman (26:55):

Thank you so much for spending time with us today and for chatting with me. It was very nice to meet you, sir.

Eric Allman (27:00):

Very nice meeting you. Thank you.

Scott Hanselman (27:02):

We've been chatting with Eric Allman. This episode of Hanselminutes is in association with the ACM ByteCast, and we'll see you again next week.

ACM ByteCast is a production of the Association for Computing Machinery's Practitioner Board. To learn more about ACM and its activities, visit acm.org. For more information about this and other episodes, please do visit our website at learning.acm.org/bytecast