Interview

M. Satyanarayanan on Mobile and Pervasive Computing

Mobile and pervasive computing open up new disruptive opportunities, such as enabling connectivity from wherever you are, intelligent personal assistants, and computation and communication ability in any device and at any location. At the same time, mobility and pervasiveness expose disconnection problems, limited battery lifetime, requirements for new human-computer interfaces for small devices, privacy and security issues, and so on. In addition, many of the traditional challenges of distributed computing become even harder to support. For example, transparency, scalability, fault tolerance, and security are increasingly more challenging if every person has a few mobile devices. The biggest challenge of today's technology is how to balance all the advantages of mobile and pervasive support with all the exposed problems. This challenge becomes even more important given the wide deployment of computer systems that now permeate into everyday life and make our dependency on them ever greater.

Satya is a unique researcher who can speak about areas of distributed, mobile, and pervasive computing with a lot of authority deriving from his practical experience. His experience goes back to the Andrew File System, representing traditional distributed system aspects, over Coda and Odyssey, addressing mobile computing, and now with his recent work in pervasive computing. Satya represents a computing history in his own right. I hope you will learn from this interview as much as I have.—*Dejan Milojicic*

Dejan Milojicic: How do you see computer science evolving? What is the next big step?

One way to predict where the next big breakthrough is coming from is to ask what the overall system's critical resource is. I think most people would agree that the answer is human attention because it is a scarce resource, especially in environments that involve some degree of mobility. The demands of daily life make it even scarcer. The big breakthrough is going to come from systems that somehow find ways to trade off hardware resources and computational resources to lower the demand on human attention.

The same point is true if you look at system administrators. The people involved in running a system are a scarce and expensive resource, and finding ways to most productively employ their time and attention is critical. So regardless of whether you take a system administrator's viewpoint or an end user's viewpoint, reducing demand on human attention is a big, unsolved problem that awaits us.

Do you think that addressing this critical human resource is required or will it continue to be observed from the outside?

I don't know the answer to that. Psychologists, especially cognitive psychologists, have investigated how the mind works, how human attention works, and so on. What is really required is that people in the systems arena like us pay much more attention to what is known and understood in the world of human cognition.

What kind of lessons can we learn from the past?

One thing that I learned from past experience, something that I never really realized when I first started, is that it's very important to provide the lowest possible usage barrier. In other words, in designing a system, you have to ask yourself, what kind of people will use this system? And secondly, how can I minimize the cost for them to try out the new system or system capabilities I'm offering? This sounds obvious in hindsight, but time and again I've seen situations where a perhaps superior overall design was not widely accepted because its designers hadn't paid enough attention to bridging the gap between users and designers.

A related but slightly different point is that you have to minimize the amount of buy-in. That is, if you have a new idea, it's helpful for potential users to try it out in small doses. If it requires programming in a new language, the amount of buy-in is very large, which doesn't endear it to users.

I also think getting early and widespread use of your ideas is key—the intersection of real users with different assumptions and you as the designer or researcher with a certain original set of assumptions is where new insights come from.

Is simplicity always the Holy Grail?

There's no doubt that simplicity is highly desirable. However, complexity is sometimes inevitable. If your system only needs to work when A, B, C, D, and E are all true, fine—implementation can be pretty simple. But if you say the system has to work under all conditions, the failure of varying bandwidth, this, that, and the other, now you've transferred complexity into the implementation. This is a tension that is inherent, and part of what makes a piece of good research is identifying the right balance.

I think it was Einstein who made the observation that things should be as simple as possible, but no simpler. And I can't find a better phrasing of the right way to think about simplicity. Yes, frivolous complexity should be avoided, and at any point in the design when you're contemplating a more complex solution, it is well worth your time to rigorously ask if that complexity is warranted. In cases where it's inevitable, I think people have to bite the bullet and accept it.

What roles do academic researchers and labs play?

For computer science to be a healthy field, there must be a genuine possibility for revolutionary ideas to emerge and, from time to time, to be successful. The field must have enough mind space, collectively, to think of new approaches, maybe those that run counter to conventional wisdom. Also, the people who want to explore such approaches need the resources and time to demonstrate and explore such avenues. Venture capitalists want to make a profit, and a contrary-to-conventional-wisdom approach doesn't always guarantee success. Hence, there is a need for government funding or other forms of fairly broad, fairly unconstrained support for exploring approaches of this kind. Research labs are typically captive to a particular organization, but the university community tends to have less vested commitments to corporate viewpoints. And that's the unique role that academic research plays.

What big breakthroughs do you see for the future?

Probably lightweight wearable computers. Continuing innovations by industry into hand-held and wearable computing technology are exciting, but they're still partially in the university labs and partially in the real world. Imagine having a pair of glasses, no heavier and no bigger than regular eyeglasses, but with a small processor in them that can communicate with a nearby compute server. It sounds like science fiction, but the technology is almost there to make it happen. I think people don't recognize what a revolution it could provide in all kinds of areas pertaining to pervasive computing.

What this sort of technology will make possible on a grand scale are applications of augmented reality. Researchers have demonstrated augmented reality with expensive hardware in labs. It's plausible, for example, to put on a heads-up display and see the Place de la Concorde as it might have looked during the peak of the reign of terror, with the guillotine in the center and so on. This opens up a host of fascinating applications. You could wear a pair of glasses with a small amount of face recognition built-in, look at a person, and his name would pop up in a balloon above his head. You could know instantly who the person is, even if you don't immediately recognize him. I look at my tree, and a little balloon pops up saying, "Water me," I look at my dog, it says, "Take me out," or I look at my wife, it says, "Don't forget my birthday!" I'm being facetious here, but there are more serious possible applications for this kind of technology.

Who do you think will drive pervasive computing?

I don't see any one party being the dominant player. I do think that the field is going to come into existence partially through research done in universities and partially through graduate students and undergraduates who participate in this work, gain initial insight, and then move on to industry.

For example, the hand-held machine or wearable computer that might cost much more than what somebody would pay for it as a commercial application is the kind of device that university researchers can use and enhance today. I'll still emphasize, though, that usage experience is critical. This is often difficult for university researchers because it is often hard to convince people to use a flaky, experimental system. You actually have to give them sufficient motivation. If the use of the system is not realistic enough, then you learn that it's not useful.

One of the big challenges is how to build systems of sufficient robustness and quality in the university environment that can provide support serious use and hence provide useful lessons. I think that this is the sort of difficult challenge that will apply in pervasive computing since human attention spans many parts of a system.

What about peer-to-peer networking and computing?

It is hard for me to get excited about it. Here's why. At the very low level of machine A and device B interacting, I think of everything as peer to peer. To me, peer-to-peer technology is certainly an important building block, but I don't see it in any way as a revolutionary thing.

What about mobile computing?

I've already mentioned the demands on human attention, and with mobile computing, it's especially critical to conserve the user's attention. Another issue is figuring out how to cope with finite energy. Should we, in anticipatory or proactive ways, guide user behavior to cope with energy concerns or should we focus on the system?

One of the lessons that I've learned is that a mobile computing environment is an inherently complex one. You have to provide greater, broadly characterized, instrumentation and better end-user training to be able to cope with it. This is something that I hadn't appreciated earlier. End users require more sophistication to cope with a mobile computing environment, which doesn't mean that on a regular basis the environment is going to demand a lot of their attention. In fact, the very purpose of many systems is to try and make the normal case easy. Edge-case handling and the number of edge cases is where mobile computing become tricky. It's like looking at a plane's instrumentation: It's much more complex than a car's, because there are more variables that matter. Mobile computing means having more sophisticated instrumentation, but we have to figure out how to keep it unobtrusive so that it doesn't demand the user's full attention under normal conditions.

I also think users are going to have to be more savvy. Until now, we've assumed that we can take a random person off the street, put him in front of a mobile computer or pervasive computing environment, and expect him to be able to use this system well. Yes, that might work when things are going well, but he should understand that the system can stall or that many other small catastrophes can happen.

So you expect that in the future, for certain classes of systems, people will have to go through some sort of school to be able to "drive" these computers?

I think that we're kidding ourselves if we believe otherwise. I think that you have to be more sophisticated in coping with those environments and here's the big paradox of computing. On the one hand, it's wonderful to have your PalmPilot or your hand-held, but the next step is moving to wireless connections, where it's talking to things and some of your bits are arriving from elsewhere. You have to be more sophisticated to understand what's going on. So, to effectively use those environments, people are going to have to become more sophisticated. One way to achieve that would, of course, be through explicit mechanisms like you just described—a school or a formal licensing mechanism. But it doesn't necessarily have to be formal—people just have to become better educated about the environment and what can go wrong.

What is your perspective on wireless versus wired infrastructures?

This involves predicting what the wireless landscape of the future will look like. If we wait long enough, will the world be covered with high-bandwidth, freely available wireless communication so that all the difficult challenges of mobile computing basically vanish and we're back in the wired world? I don't believe so, and the reason why is a fundamental limitation of physics.

If I want more bandwidth, I lay more fiber. But you can't do that with the radio spectrum. At best, you can reduce cell size. The instant you have that kind of limitation, it requires investment of real resources such as base stations and all the associated maintenance and so on to actually make wireless communication work. This means that it is very unlikely that wireless communication is going to be free.

So, the world I see in the future looks like this. I see a big desert with many oases, and inside each oasis, life is wonderful. An oasis might be a campus, a department, or even a city. Basically, inside that environment, there is enough incentive and enough business opportunity or enough investment by the community to offer high-quality wireless communication that works extremely well. But beyond those limits, you either have poor, low-bandwidth, or expensive coverage along some dimension of quality. As technology improves, we can imagine those oases growing or even merging together, but certainly in the next 10 or 20 years, I don't see the model of high-quality, high-bandwidth, almost free communication everywhere. I really don't know if it'll ever happen but I certainly don't see it happening in a decade or two. On the other hand, wired communication is a different story—it will always be important for the fundamental reason that you can always get more bandwidth by laying more cable. You don't have that trade-off in the radio spectrum.

What are your thoughts on standards?

Standardization is a double-edged sword. It allows multiple implementations, fosters all kinds of competition, and so on, which generally benefits the public, but every standard you make represents a frozen point in a technology's evolution. As technology moves on, that frozen point could become a burden rather than a benefit.

I think we're better off when standards can emerge through intrinsic technical merit rather than through political process. The big danger is that industry-driven standardization tends to be short-term, which is unfortunate. Universities or research labs can help in the early stages of standardization where you build things to evaluate their technical merits with the full knowledge that they might be discarded. I think explicitly accepting throwaway systems and then making sure that you can capitalize on your learning in the creation of standards is the best way to form standards. The problem in industry is that having put all the time, energy, and money into building a prototype system, throwing it away is very hard to justify.

Can you give us some important lessons you learned from your vast experience in the computer science field?

I think we should look at the entire open-source experience's evolution. The ability for many users to play with Linux and improve part of it was an important part of its success. If you go back, say, 15 years ago, there was a culture of system building that believed the experiment wasn't complete until the system was used in practice. My own experience has been that this notion of validating experimental computer systems through real use is critical. I cannot emphasize it strongly enough. You have to find people who do not share your original assumptions or are neutral, explain your system to them, give them the opportunity to use it, and then see what happens. This is the best validation because it's where most surprises happen. This is also an enormous amount of work because, to get to that point, your artifact's quality has to be pretty good. If you're going to have somebody drive a new car, it has to be road-worthy. Putting the sweat in to get to that point is more than many workers in academic and research labs today are willing to do.

Another lesson I learned is to take a long-term perspective. It's amazing thing to me how long things take! Even after you take into account all the unexpected things that you think you might run into, things still take longer. The net effect is that anything you build that is even remotely interesting and useful often has a longer life than you suspected. So, make sure that for critical short-term design assumptions, you ask yourself if your product will be interesting five or 10 years from now.

What advice would you give researchers just starting their careers?

It's very hard to give good advice because it's very dependent on your particular interests and what you're trying to do. Here are some very broad and general suggestions. Know yourself. Pick a problem that's worthy of you. Don't settle for a problem that looks easy. Don't try explicitly to pick a fashionable problem.

The best insights come from people seeing things that other people didn't see. Research is very much like trying to see faint stars in the night sky. If you want to see them, you have to get away from the city. Similarly, being surrounded by groupthink is a sure way not to notice the little things that ultimately make a big difference. We all agree that close ties between industry and academia are important, and I would certainly agree with that 100 percent. But you have to be careful not to be so influenced in your thinking that it's like being in the city and not being able to see the dim stars.

Mahadev Satyanarayanan is an experimental computer scientist who has pioneered research in the field of mobile information access. One outcome of this work is the Coda File System, which supports disconnected and bandwidth-adaptive operation. The system he's currently working on is Aura. He is also the Carnegie Group Professor of Computer Science at Carnegie Mellon University. He received his BS and MS from the Indian Institute of Technology, Madras, and a PhD in computer science from CMU. Contact him at satya@cs.cmu.edu.

Originally published in IEEE Distributed Systems Online magazine, September 2001