CHASE          FOX          FIGUEIREDO          GRIMSHAW          WATSON

# Thoughts on the State of Cloud over the Next Five Years

**Jeff Chase,** Duke University
**Geoffrey Charles Fox,** Indiana University
**Renato J. Figueiredo,** University of Florida
**Andrew Grimshaw,** University of Virginia
**Paul Watson,** Newcastle University
**Mazin Yousif,** T-Systems

*In this issue of* IEEE Cloud Computing, *EIC Mazin Yousif talks with experts from US and European universities about the current state of cloud computing as well as where the technology is heading over next five to 10 years. They cover critical cloud topics, such as security, privacy, and standardization.*

**Yousif:** We're going to start with few statements on the current state of cloud, then dive into specific cloud topics such as technology, security, privacy, and standardization.

**Chase:** With respect to the state of cloud computing technology, why don't I start? When I think of cloud computing, I really think of hosting providers. The National Institute of Standards and Technology [NIST] has given us a pretty good definition of various levels of cloud computing. I think we're all familiar with the terms "infrastructure as a service" and "platform as a service," and that's primarily what I mean when I say cloud computing. Of course, in the popular press, cloud computing is often used to mean server-based computing. This is the era of megaservices. Facebook and Twitter and so on serve hundreds of millions or even billions of users, and these run in cloud datacenters and they have a massive scale that we could not dream about a decade ago. They do support some application features and add-ons, but they're not what I think of as general-purpose cloud computing environments, although they are legitimately part of this discussion.

The true cloud computing hosting providers are seeing increasing adoption. There are a number of players in the market now, a few very large players. It's definitely mainstream. Lots of new companies have gotten their start on cloud hosting systems. Sometimes it's hard to know who's running on the cloud and who's not, but when we have these Amazon Web Services (AWS) outages and then you can see who goes down, right. Pinterest and Netflix, and other big-name services that people use very frequently, have parts of their operations running on clouds.

So this adoption is increasing. But one of the major barriers is the confusion over security. People don't always know if they can trust their cloud provider, if they can trust the provider to isolate them, what additional vulnerabilities they might be exposing themselves to by using the cloud. And I think that the Snowden revelations—even today, I first saw a report that information is coming out about what steps the US government is taking in working with private companies to be able to essentially spy on their customers or traffic carried over their networks.

And I think partly as a result of the confusion introduced by all of that, we very well may see a flowering in which there are many more providers entering the market. You can consider it fragmentation or balkanization or reduction in economies of scale or a flowering of additional options, but I think we're going to see increasing diversity in the market. And that, of course, makes the question of cooperation and integrating different kinds of cloud services an important issue. Just to sum up, I think we're seeing a lot of diversity in terms of the services offered, a lot of uptake, and a lot of interesting questions having to do with security of data in the cloud and security of data sharing in the cloud.

**Grimshaw:** I agree with a lot of the things that were just said. To me, like a lot of new technologies, there's a lot of hype right now and a lot of people claiming to be cloud, and I think that the NIST taxonomy is a good taxonomy, and I think in IaaS, it's very well understood and people are going forward. I take sort of a broader view in the sense that I've been doing distributed computing since the very old days, and there are a lot of similarities to the challenges and problems that people are facing now in cloud are the ones that we've faced for a long time, particularly as you crawl your way up the stack. I agree that it's not just service-based computing because, to me, that's just the same old 1960s era, I have a very big machine and people come into that very big machine. It's ignoring the challenges of how do I get a number of these different services to interact, how do I manage federation, how do I manage delegation and security across that environment.

Another way for me to think about it is that I just look at cloud computing as being distributed computing where we give people different interfaces into different levels of a stack. At sort of the lowest level, we have, "Well, gee, we can start you a virtual machine." In fact, in some places, I can give you some bare metal. I can give you a VM with your choice of operating system. I can give you some higher-level services that start a job or access a file or exchange one credential for another credential, all the way up the stack to essentially full up application services. So to me, when I look at all this, I hate to sound boring, but it's a lot of the same old distributed systems problems that we've had for a long time, and

one of the things I personally would like is, as people go forward hyping and saying that they're doing cloud things, they at least look back 15 or 20 years through the grid work, the distributed systems work, metasystems, things like that, and say, gee, has this problem been solved before. I've been seeing a lot of cloud papers lately where I saw the same paper 15 years ago for grid and 10 years before that in metasystems. So that's sort of my take on that.

**Figueiredo:** I tend to agree. My view on cloud computing has always been from the infrastructure as-a-service perspective and I think the NIST terminology helps broaden that scope a little bit, but the use of the term, like Jeff said, in popular culture, has been just stretched a bit too thin. As Andrew said, we're looking at problems that reoccur—time sharing and implications in distributed systems—and what's really interesting here are the models of cost associated with cloud computing and the granularity of resource allocation, and the charging of resource allocation. So the fact that you can have virtual machines with prices that fluctuate over time that you can buy and release on a short notice and with the granularity of hours, that's, I think, what really has made the cloud computing model quite different from things we've seen before.

Now, the security implications, as Jeff brought up, are still issues we are struggling with, and the models will have to evolve based on what people expect when they start using these systems and when they realize that the guarantees they may get from a provider are not sufficient for what they would like to have for the system they're outsourcing to the cloud, if you would. At the end of the day, the problems of multiplexing, isolation, and resource management at the IaaS layer are at the core, and have been addressed with the help of virtualization technologies. I think IaaS-layer problems have been addressed to a large extent, and now the interesting problems become more of management and those that arise by going up the stack: services, interoperability, and being able to tap into multiple providers.

**Watson:** I think the interesting thing about cloud is as much the business model as the technology. I mean the way in which you can grab computer resources when you need them on a pay-as-you-go basis. Because of this, cloud has real potential to transform a lot of organizations, but it's at a very low level at the moment because of this emphasis on infrastructure as a service, so what we're seeing is a lot of the organizations, and a lot of the researchers that

have the potential to benefit most from cloud can't actually benefit because they don't have the skills to build the scalable, dependable, secure distributed systems they need. These are difficult problems confronting anyone building large-scale systems, and clouds don't solve those problems. You've still got to have people who can build out systems with those characteristics and there are very few people around who can do that.

So we find all of these organizations that could benefit but can't actually benefit at the moment, except that sometimes you do have early adopters, sometimes forward-looking organizations, sometimes heroic individuals that can build out systems on the cloud. But what I'm seeing at the moment is that this is creating some tension because organizations have particular sorts of governance that they want to conform to in terms of reliability, security, and audit trails. For example, pharmaceutical companies are worried about passing FDA audits, and security's a general problem for all organizations. And then you get these heroic individuals who go out and build out services in the cloud. In the old days, they would have to go through the IT department and, okay, it might take them a lot longer, but at least the IT department would have made sure that the government's processes were followed, but these individuals aren't necessarily following the organization's governance and sometimes companies are getting into trouble because of that. So I see there's a problem at the moment in realizing the benefits of the cloud given the difficulty of building scalable systems on a cloud, and the need to maintain a level of corporate governance.

**Fox:** I would agree with a lot of what's been said except I think I'm more optimistic about the current state of cloud. I also think clouds have made tremendous innovations, not just as the infrastructure as a service, but at what's usually called platform as a service. I think technologies like NoSQL, MemCache, and MapReduce exist independently of the actual cloud business model, but they were created by the cloud revolution, so they're usually grouped in the cloud bucket. I consider these to be pretty important. But in general, I agree that key features of clouds are the hosting services and the business model. Further I expect clouds are likely to evolve to have larger use. One of the reasons that I think they're not so much used at the moment are performance issues due to virtualization and low performance in networking and things like that. Those performance degradations, I think, will tend to disappear.

If you look at the research use of clouds, at least in the US, it's pretty small whereas the commercial use has sort of exploded; I don't think the same is true of the research use, and I think that's an area that has some potentiality to change because I think the clouds are cheaper than other solutions. And that's going to put pressure on these other solutions such as university computer centers, and I think we'll see changes in that regard.

**Yousif:** Cloud is not always cheaper.

**Chase:** I was just going to comment on the economics and say that the value proposition for hosted cloud computing depends in part on the elasticity of demand. We have this pay-as-you-go property that cloud computing gives us, so if your demand is very bursty, the economics for renting machines are great. But if you have a steady, sustained long-term demand, you may be paying a premium for somebody else to manage your infrastructure.

**Grimshaw:** I would agree with that. I'd like to get back to the "pay-as-you-go is fundamental." But that's, of course, been around for a really long time in lots of forms. I mean, when I was an undergraduate, which was a long time ago, we had to pay per use, so I think that it's just one of the things that people must think about.

I'd like to return to virtualization. Everybody behaves as if virtualization is a new thing. You know, we've been doing virtualization for a really long time. The IBM 360, if I recall correctly, could virtualize instruction sets on the fly, and the whole notion of virtual machines as a way of handling heterogeneity and to build layers of abstraction. That's how you build systems, by layers of virtualization. So to some extent, I think a lot of it is hype. I think the VM, the hypervisor virtualization has been tremendously successful, and that's a technical innovation but, you know, one has to ask why people feel the need to do that. A lot of it's because of isolation. They didn't trust the operating systems to do the right job.

I was talking to Doug Thain [www3.nd.edu/~dthain] about a year ago and he said that perhaps the reason VMs are so popular is because the operating system community failed to provide the things that users wanted in terms of process isolation and the ability to support legacy codes.

**Yousif:** Yes, virtualization has been there for quite some time, and the basis of visualization is exactly what was done with mainframes in the 1960s and 1970s.

Going back, I heard a lot about IaaS and PaaS, but when you talk about a cloud, you talk about services from all layers of stack. The question I have is what layers you think provide the most value for the consumer (for example, enterprises, academic institutions, scientists, and HPC [high-performance computing] centers).

**Grimshaw:** One of the problems in the scientific community, I heard that they tried charging at the national centers years ago, and the problem is that researchers oftentimes will choose to spend the money on something else, and so the question in sciences is, if you give people a blob of money and say you can spend it on the cloud or you can spend it on another graduate student, what will they do? I mean, that's one of the reasons why allocations can't be converted to money at the national centers. At least that's what I've been told. So I'm not sure scientists like the idea of actually paying money for cycles.

**Chase:** I think there's a great opportunity for virtual currencies in this space, to create a closed economy.

**Grimshaw:** I agree 1,000 percent there. Yes, I agree.

**Chase:** And the question of how to manage such a virtual currency, I think, in and of itself is an interesting research problem, but there's been a lot of progress in the computational economics to be able to do market-based resource management. I know that the day is going to come when we can do a good job with that. We're not quite there yet.

**Grimshaw:** I think (and actually, since we're supposed to be talking about are we into the future yet or are we still on the current) that will be an area where, at least in academics in the next five or 10 years, you will see a tremendous amount of move toward providing virtualized currencies for researchers to spend however they want on computational infrastructure and in having institutions trading each other for resources, whether it's cycles or storage or running an application so that they can just get their work done and decouple it from the, "gee, I have a cluster in my lab and I have to load everything on it." I think that will be here in five or 10 years easy.

**Yousif:** That's actually a good point to jump to. Paul, Geoffrey and Renato, do you have any comments on what we just discussed?

**Fox:** I wouldn't necessarily agree with that. I think it's rather implausible that NSF [US National Science Foundation] will announce its federal budget of the year as dollars X of real money and dollars Y of virtual money because virtual money is real money, is it not? So I think a model where you have virtual currency could be part of the solution, but in the end, it's going to come down to real money. And let's come back to the economic issue. I'm part of a collaboration, and they tell me we can't use cloud because it costs money to store data on the cloud and my data storage is free at my university. But that's, of course, my overhead that's paying for that free storage, so there's a lot of rather complicated issues about economics with virtual currencies.

> I think we're in a position where hypervisor platforms are pretty stable and will keep making improvements.

**Yousif:** Let's address the economics topic later. Let's jump in now to the second part of the roundtable where we look at specific features for cloud and then discuss how you see them evolving and what maturity means. Let's look at cloud technologies like virtualization and manageability.

**Watson:** I think that the main trend is going to be one that Geoffrey hinted at, which was about a move up from infrastructure to platforms. My hope would be that by 2020 the number of organizations and researchers that can use cloud will be greatly increased because there will be high-level platforms available that they can use, that already provide a lot of the characteristics that are really hard for a typical programmer to deliver, such as scalability, security, and dependability. We're already seeing scalable data storage as a service being provided so you don't have to worry about how you build systems to achieve that yourself, and I'm expecting that there'll be a range of application servers, data analysis servers, and webservers that will be offered as a service by cloud software providers. So I think an interesting change for the industry will be that, if you look at a typical software company today, they produce either software that you can download and deploy on your own servers or they produce something that's open source, which you can take and do what you want

with, but I expect that we'll see a move to companies producing their software as ready-packaged services in the cloud. By 2020, I'd expect this to be the main method for delivering software to users.

**Chase:** I agree with that. I think Geoffrey actually put his finger on a lot of this, that certainly the development is going toward less and less of a cost for virtualization and more platform features and higher-level programming features as Paul said. But also, another big thing that's happening is that the availability of high-powered, highly scalable software tools to deploy into a cloud, into the resources that a tenant controls in a cloud, what we would call a "slice" in GENI [Global Environment for Network Innovations], is really expanding. Somebody mentioned the MemCache service, which is heavily used in Facebook and in other systems as well. I think one of the things that's happened is that a lot of the research work in the distributed systems community on peer-to-peer systems and on building highly scalable systems using key-value stores and distributed hash tables has departed from its original context of networks of PCs, but has turned out to be very powerful in the cloud. I would count that as a success for the research community. And these tools are enabling construction of scalable applications. And cloud, of course, is an enabler for that because it makes the infrastructure available to deploy onto.

**Figueiredo:** I think that, on the virtualization side, we have had, for a decade and a half now, steady improvements, and I think we're in a position where hypervisor platforms are pretty stable and will keep making improvements. I think virtualization software is in pretty good shape now, and the open source software that runs on top of it and manages virtual machines is still churning and still complex to deploy and manage—for example, OpenStack and the software that you lay on top of that. They're very powerful but still not easily accessible to the average user, so I think you just have to wait for the technology to mature, and I also agree that, as we move forward, users will be more interested in the platforms as a service deployed, but they will not necessarily be provided by the infrastructure from the service provider. I think there'll be room for middle-tier services that might even be able to tap across multiple providers and select the proper one and sort of hide a lot of the low-level IaaS headaches from the user, but still use IaaS underneath as the vehicle for deployment.

**Grimshaw:** I think you're going to have a market form in which you have layers of service providers providing different types of things. There will be IaaS providers, and their margins will likely get thinner and thinner and people like Matlab—right now you can use Matlab as a service. There's going to be a lot of these—this XYZ as a service, and they're not going to host their own stuff. They're probably going to host maybe some of it if they have a base load they can count on, but otherwise use the infrastructure providers to do that. So I think, like any other industry, you'll end up with essentially layers of providers, where at the top end you have the software or the experience or whatever you want to call it, that the user is going to be the actual person who's paying the money at the end of the day, and then that will go through a series of markets and industries down until you have the infrastructure providers who actually run the giant machines and get the power set up and all that. So I think we'll be like many other industries in that sense.

**Fox:** So sort of following the earlier conversations, I think an exciting feature of cloud today is the open source software development, which includes things like Hadoop and Hbase. That appears to me to be somewhat driven by clouds because it is used a lot by the users of clouds, and so we're actually benefitting enormously from these software projects in the data analysis area. I call this the Apache big data stack. I made a list of all the projects I could find in that area, and there are more than 100 software activities in that broad area (about third from Apache), and that's a really important development, in my opinion.

**Grimshaw:** Geoffrey, do you think that's because the software licensing problems are not there? Certainly I do because you don't have to worry about licensing. You're not going to be breaking the law if you start a new version, or violating a copyright. If the licensing issues were dealt with, do you think more people would use stacks that were provided by commercial vendors?

**Fox:** Well, that's an important point, but it may be that products with a difficult license model will be forced to change by competition as there will be better and better open source products or more models like Cloudera or Redhat with commercial value-added on a base open source platform. I'm pretty certain that open source software is going to be a winning feature of the platform and software as a service we see in a few years' time.

**Chase:** That's an important comment. I think it applies all the way down the stack. Renato mentioned earlier the enhancements in the lower-level infrastructure management technologies that are open source. It's worth noting at this point in the conversation that we do have software systems for cloud hosting that are open source—OpenStack and Eucalyptus and others—and although there are still some kinks to shake out, they're suitable for anybody who can buy some servers to set up their own private cloud. A lot of organizations, including my university, are doing this internally as a way to meet demands for more fluid access to computing resources within the enterprise. That's a big part of the story here, and a lot of that is being driven by open source.

**Yousif:** So there are several angles that you touched on in the discussion. The role of open source, which is something you're going to need to address later.

We have tons of software running in datacenters in IT environments that might not be suitable to run in the cloud. What do we see in terms of what is required for software for an application to efficiently run in a cloud?

**Grimshaw:** I think the premise of the question is a little strange. I mean, software currently runs in an environment. Certainly it should continue to work "in the cloud" if it's on a virtualized piece of hardware and an operating system. I think the more interesting question is will new applications that interact with multiple components and are not silos the way most applications are now, what's the interaction paradigm going to be between pieces of software going forward? Is it going to be REST? Is it going to be SOAP? What are the basic interaction paradigms and the architecture, and how are security and delegation and accountability going to be done in that kind of environment? I think that's the right software question. Current software that works on a standalone machine is going to work in the cloud on a virtual machine.

**Figueiredo:** I could add a little bit to that. I think virtualization has improved over time, but it's still not trivial to take a workload that you have locally and deploy it to the cloud. There are still differences in hypervisors and formats and the idiosyncrasies that make deployment complicated. Ideally you'd like to take the machine you have now and deploy it in any cloud provider and actually be able to move it around if the price is better somewhere else, or if you want replication and to avoid vendor lock-in, but the mechanics of it are still quite cumbersome. You

have to create different images. They have to deploy in different systems, even within the same provider, and different zones, and you have to worry about that when creating images. So I think that a lot of the promise where virtualization is totally transparent to the user is still not realized. We were talking about technologies and services. I think one of the technologies that's showing promise here is the ability to do nested virtualization; in other words, run VMs within VMs. Now you have all different flavors of that: you can have lightweight VMs like containers running inside VMs, as well as other VMs. I think because cloud providers don't necessarily agree on the interfaces and what's exposed to the user and lack interoperability, it's still difficult to tap into multiple providers. Raising one level up with nested/recursive virtualization, I think provides an interesting potential to allow users to define what they want and just use the lower-level IaaS to deploy sort of a good "trampoline" environment for what they really want to deploy.

**Watson:** I agree that there's the opportunity for lots of innovation in the cloud. There's the potential to move all sorts of existing software in the cloud to make it easier to meet the needs of system builders and end users. I think what's going to be important is to have organizations that might not be the developers of the software in the first place, but who can take software, for example, open source software, and package it as a supported, well-managed, scalable, secure service in the cloud because I think, for the reasons we just heard, it's very difficult for a typical programmer or user, whether the person is working in a company or is a researcher, to do that. I think these intermediate organizations producing these scalable services will also be able to deal with issues relating to the heterogeneity of cloud infrastructure. They'll be able to provide users of the software with exactly the same service on different clouds even though the underlying APIs might be different. I think these "cloud packaging and support" organizations are going to become really important.

**Grimshaw:** So, Paul, you're basically arguing, and I think I agree, that even the IaaS is not necessarily the right level of abstraction, that what people want are higher-level services that they can interact with and not care about the back-end layer. Essentially they want to enter at a higher level of the stack than a virtual machine.

**Watson:** That's exactly what I mean and I think that's the direction the clouds will move in; other-

wise, cloud software development will always be restricted because of the lack of skilled programmers who can actually work at the IaaS level. If clouds are to get out there to impact the vast majority of organizations and the vast majority of researchers in universities, we clearly need to allow them to move right up the stack, building systems from scalable services already provided in the cloud. And so we need people and organizations with the skills to design and build these high-level services that others can then use.

**Yousif:** So if you look at the industry, I can see a move to deleverage IaaS a bit and focus more on PaaS as well as SaaS [software as a service]. There's a lot of discussion now in the industry that in the next five, maybe 10 years, enterprises will likely follow the SaaS model. They will not be buying software and paying licenses and managing it internally. Any comment on whether that's really how we're going to see it going forward?

**Grimshaw:** As an academic I think the second one of those, "how do we build composite services?" is much more interesting, because internally an organization that's selling a particular service will just do whatever it needs to do. But if I want to be able to build mashups or whatever word you want to use, then it becomes important that we have standards, standard ways of saying how am I going to interact with this service. How am I going to pass authentication and delegation information across that? How am I going to pass information about how to charge me in a way that is traceable and auditable and all of those kinds of things? And I'm a little concerned with some of the way industry's going right now with this rush to basically say, well, we'll jam it all in as a REST document and let people figure it out. As a software engineer, I like to know the interfaces are there that are defined.

The question is what the interoperability paradigm is going to be. Is it going to be typed or not typed? I'm a big believer in strongly typed systems.

**Fox:** One comment on software as a service. I think in science, at least, there are quite a lot of recent successes in so-called science gateways like chemistry or biology simulations as a service. So those are essentially software as a service. I think that is successful and you will see lots more of it, whether it's called gateways or software as service, we can debate.

**Chase:** It makes applications available without the user of the application having to manage it at all,

which is really compelling from the standpoint of users as well as all the elasticity benefits. I just want to add that, with respect to the original question, independent of the software-as-a-service trend, software distribution, even for software that runs on consumer devices or user devices is also cloud-based. So we've increasingly gotten to a place where the provider of the software is—you're connected to it and it's running on a service backbone and software updates might be completely transparent to you, and, of course, that also means that a similar benefit to software as a service is that, from the standpoint of the provider, they can control the software distribution and can shut it off at any time as well, so increasingly we're moving toward a model of recurring revenues, more frequent upgrades, and faster obsolescence for somebody who blocks the upgrades, and that's probably good from the standpoint of revenue for software providers.

**Yousif:** One of the challenges for security in general is that security's requirements go beyond technology. For example, there is a need for comprehensive processes and governance around it. Service providers usually do not publish their internal processes or internal governance when it comes to security. We have no idea who has access to our data in cloud providers' servers. So how do you see organizations fully putting faith in the cloud when they don't have that visibility, and if they don't, what do you think needs to be done to make them trust cloud providers?

**Grimshaw:** Obviously you can't make them trust, so the question is how you increase the level of trust. A lot of problems—I don't want to overgeneralize— boil down to indemnification, and that's, I think, just going to continue to be an issue. Can I trust somebody to handle my data or my whatever if, when they screw it up, I (a) am not held accountable, but (b) have the ability to recover from them compensation for whatever the problem was? And right now, I just don't see the risk management necessary in the kinds of agreements that I've seen, which is not necessarily universal, to handle some of those things. People are very risk-averse. Some people are very risk-averse, particularly as you mentioned, Germany.

**Fox:** I would say a lot of it is due to tradeoffs, mainly you're trading off functionality versus privacy. As a personal note, we were once told that we shouldn't use Gmail or Google Docs because of Google's policies about keeping things and fear about what they could do with the data we put there. Those products are however broadly used because they're high qual-

ity. Another example where I may be wrong, is the area of health data, where there are obviously a lot of privacy issues. My impression is that health research is seriously limited by those privacy issues. So we might find, for instance, that in countries like China where there aren't so many privacy issues, people might get healthier, and that might cause some reconsideration of the importance of privacy—again a tradeoff: this time privacy versus being healthy.

**Chase:** Larry Page (of Google) was quoted in the *New York Times*[1] a few days ago making a similar comment—that there's a lot of potential benefit for big data in the health space but it's hampered by privacy concerns. So I think the technology for managing authorization and controlled careful sharing of data sets is an important area for development in the future. In fact, cloud models can be very helpful there.

**Yousif:** One of the reasons for the question is to see if there is a role for auditability here, trying to audit the service providers?

**Chase:** Absolutely. I mean, there are standards for that taking shape. Cloud Security Alliance, for example, has had a big effort in the space for a few years. Increasingly, we're going to see standards for what cloud providers should be doing to control risk. And then audits of those standards that can help to build trust in third-party providers where there isn't already a direct trust relationship between the customer and the provider.

**Grimshaw:** When you use the word "standards" in this space, I think it's important to note that there really need to be standards and not just the standards we're defining, what the SLA [service-level agreement] is going to be, but also how you negotiate SLAs, and there's been a lot of work there.

**Yousif:** What do service providers need to do to enable auditability?

**Grimshaw:** It's not just the service providers, but to audit that the user actually requested Amazon to do something. I think one of the challenges is how do you have transaction chains that cross multiple boundaries where you have strong authentication all the way across and strong repudiation all the way across. Many of the models that are being used right now make it very difficult to know who actually initiated some action more than one organization away, and as long as we stay in that space where there's no delegation and no traceable, auditable delegation,

it's going to be harder to address the questions you've had. Yes, I need to be able to audit the provider, but I also want to be able to have the provider be able to prove to me that they actually did something on my behalf as opposed to somebody else's.

**Yousif:** So my question is, what needs to be done specifically in the cloud architecture?

**Grimshaw:** This gets back to something I said earlier. You have to define what your interaction paradigm is going to be between different components, and part of that definition of the interaction paradigm needs to be how identity and credentials are passed and how they're delegated and how it's done in a provable, auditable way. I mean, we've been dealing with just transport in the security architecture for some time. A lot of the current architectural styles that are being used make it difficult to do delegation because it's assumed that the client is going to be doing all the interactions, and if you call some service to do something on your behalf, right now it's actually the service that's going to interact with other services for the most part, not as you because there is no delegation model, and I think we need good, secure delegation models.

We have good delegation models. We need them to be adopted in a standard way because if there are 10 different delegation models, I can't delegate easily. There are lots of them out there and we need to pick one and say this is how we're going to do it.

**Chase:** One of the things that came out of the GENI experience is that there is a lot of need for tools for managing security, and there's a lot of research in declarative trust management from a decade or so ago that actually turns out to be extremely valuable for managing these kinds of auditable delegations and nonrepudiation with rich models of specifying what's in a credential and then reasoning from those credentials using logic. I think that's very interesting.

**Grimshaw:** Yeah, there's been a lot of work, and I think a big problem now is almost all of the industrial-scale work is using transport-layer authentication and security instead of message layer, which I personally think is a better place.

**Chase:** I agree with that. Signing is nice for nonrepudiation. Actually, while we're on that theme, if I could just make a comment about the original question about how to do auditability, I mentioned the Silver project that I'm working on with some other folks at Wisconsin and UNC-Chapel Hill. One of

the ideas there is that cloud providers will play a role in doing some attestation and/or auditing of other providers further up the stack. Renato mentioned that we might have service providers spanning multiple infrastructure-as-a-service clouds and federated clouds. And in that kind of a setting, where you're layering more service providers on top, you can have the lower-level providers observe what those service providers are doing in some elements, perhaps what code they're running or if they're running standard software and so on. They can make authenticated statements about that, and others can rely on those for building trust. I think that's a very interesting area going forward.

**Watson:** I just wanted to make a point that if you look at auditability, there's an analogous problem with doing science in the cloud because if we want to have reproducibility, then we need to have a good audit trail. The problem is that to achieve this, organizations will need to know exactly what services were used, what data was used in those services, and what versions of those services were used. However, while we talk about the advantages of moving up to a platform level, the platforms that are offered by third parties are black boxes to which you give some data and they then produce result. But the owners of those platforms might upgrade them, they might "improve" them and so the next time you use one, you might get a completely different result for exactly the same data, which destroys your reproducibility. Therefore, I think it's really important for science in cloud that we think about these reproducibility issues, and offer the ability to keep all versions of services so we can always go back to reproduce exactly the computations that we did in the past. I think that's analogous to the auditability that you need for commercial applications.

**Yousif:** Let's move to cloud standardization. There are ongoing efforts to standardize cloud interfaces. There are consortiums and academic efforts. How do you see this evolving? How do you see it becoming something real, and what do you see is the need for major service providers to put their full faith in standardization and push it forward?

**Grimshaw:** I must start with a biased disclaimer as I'm the chairman of the board of the Open Grid Forum, which is one of these organizations you're talking about. And just so that we're clear about that, standardization is a tough and difficult process because it requires people to come together and com-

promise, and that can be really difficult, particularly when there is a major player. What's their incentive to standardize? If they already have 80 percent of the market share, standardization can only help them cannibalize their own business.

So I think it's important, though, because the benefits to standardization are huge for the consumer of whatever the service is, and so we have some de facto standardization right now. You know, Amazon EC2 is sort of a de facto standard. There are standards right now at the lower layers, at the virtualization and slightly higher layers such as OCCI [Open Cloud Computing Interface], and given the broader definition of software as a service that we were talking about earlier, there are actually a lot of standards in place, open standards at higher layers such as how do you take one set of certificates and convert them or transform them into another set if that's what you want to do or if I want to run a job or access data. So there are higher-level standards. The challenge is you can have as many standards as you want, but if people don't adopt them in their products, they're kind of pointless. And the only reason a company, in my experience, will adopt a standard is if their customers ask for it, and the customers have to ask for it and they have to believe that it gives them the ability to switch vendors. So I think the biggest advantage of standards is that people can write to a standard and not have to rewrite for every platform. Also standards change more slowly than vendors typically want, so when you have it, you can avoid vendor lock-in.

**Yousif:** If you are an 800-pound gorilla now and if you start losing market share, then you will probably be more in favor of standardization. Any other comments?

**Grimshaw:** As long as it's standardized on your stuff. I work with a lot of companies that are like, yeah, we'd like to standardize. Here are our interfaces. And by the way, we want to control the standard going forward. I mean, wresting control of the standard away from a single stakeholder is important because otherwise the other stakeholders aren't going to be interested.

**Yousif:** Also talk about the role of interoperability among those providers when you talk about standardization as well.

**Grimshaw:** I think it's critical. Look at USB sticks and how that standard has helped us all.

**Fox:** I think standards will increase in importance, but I think we need to settle on the levels (APIs) that one can usefully standardize. Technologies such as SQL have an active standard activity, but that's a relatively mature and well-understood technology. If you go to other technologies that impact clouds, the immaturity and rapid changes handicap standardization. Let's take a simple important example, Hadoop or MapReduce. It's not the correct time to standardize this because it's not done quite correctly at the moment in my opinion. It will change significantly and evolve, and so we should wait a bit. There is another comment I wanted to make, I think the concept of software-defined systems, which sort of generalize software-defined networks and things like that, will grow dramatically in importance.

Note that software-defined systems require standards. You can't really define something in software unless you have an agreed-on language to define the system components. Reproducible computing is one goal that can be based on software-defined systems; we need standards and ontologies to be able to define a computation for perpetuity. However, we know some standards didn't work out. For example, the area of Web services shows many standards that did not get adopted. I put lots of effort into Web service reliable messaging (WSRM), which was, as far as I know, not terribly successful. BPEL [Business Process Execution Language] is a workflow standard that's also had less uptake than initially expected. Standardization is, as Andrew said, very hard. You need to do it at the right time when we can identify the correct layers of the architecture to standardize.

**Figueiredo:** I would like to add that I think standards evolve at a slow pace and, again, having one major player that dictates a large share of the market doesn't help. But, at some layers, I think you begin to see some convergence. For example, you have the ability to create virtual machines in the use of libcloud and libvirt, and if you're able to use different providers to deploy virtual machines, you start seeing systems that layer on top of these services. For example, I've worked in collaboration with a group in Europe, and they have a platform-as-a-service system (ConPaaS) that is able to talk to multiple back-end infrastructures using these interfaces. You don't have a standard, but you have a small enough number of different systems that you're able to manage to some extent the ability to deploy virtual machines on different providers. So I think you see some efforts that will be working around the lack of interoperability, the lack of standards, but then at the end of the day, like Andrew says, it becomes something

that the user has to demand, and perhaps to get to that point, you need to have solutions in between that allow users to manage these differences, given our lack of standards and lack of interoperability.

**Chase:** Definitely. I think a huge element of that answer was the idea that, as you add more and more services higher up the stack, the interface that you're presenting to users or to programmers is higher up the stack, and there's an opportunity for that software to bridge differences lower down the stack. So that's the principle at work in libvirt and libcloud, which have been very successful at, for example, allowing open source private cloud systems to run on multiple underlying virtualization systems with a minimum of fuss and bother. So it's still a burden for somebody to have to pave over differences in the APIs, but they can then offer a higher-level API that isolates others from that burden and challenge.

**Grimshaw:** So the irony of that is this bridging gets done. I've been in distributed systems for a long time. Continuously we have these—name your favorite things—start a job, do this, do that, which essentially provides a virtual machine, a virtual abstraction to transfer files or run jobs or whatever, and then map to lots of different implementations. I mean, the one that was just mentioned, in the "grid world" there are dozens, and every application built their own one of these. There's something called SAGA in the grid world that does this, and then lots of different systems have built their own; they've built the same thing. And so this drives me back to a set of papers in 1992 around heterogeneous distributed computing. There was a special issue in *Computer* on how do you do that, and I think David Notkin was the editor. [*Editor's note: This refers to the late David Notkin at the University of Washington, and the work he did on heterogeneous systems in the late 1980s.*] I could be wrong, but they basically laid out this picture that says what you really want to do to solve this heterogeneity problem is to create an abstract virtual machine for X or Y service. That's what libvirt is doing, I suspect, and that's how you manage heterogeneity.

**Yousif:** How do you see the role of open source going forward when it comes to cloud? Open source has been playing a very big role in increasing the adoption of cloud. Will it increase adoption? Will it be a force for standardization? How will service providers differentiate themselves?

**Fox:** I think open source efforts will increase and at the moment there is quite a bit of pressure to join the open source efforts because they are so hugely successful. Of course not all open source projects succeed. I once worked on one, an Apache project that sort of died and we had to abandon an effort built around it. That was pretty annoying, but other Apache projects and other open source projects didn't die and they proved successful. I expect the success of open source to continue. But there's not just one open source solution for any given system. For example, at the programming level in clouds, you have many open source projects—Hadoop, Spark, Stratosphere, Twister, Hama, Pregel, and Giraph.

**Watson:** I think that if you look at what's been going on in universities over the last 20 years, there's been an awful lot of really good open source software that's been produced, but actually very little of it has taken off and been reused, either within the universities or by industry. Where it has succeeded is when there has been a commercial organization that has supported it, provided training materials, and provided high-level management on top of it. Traditionally, people in universities don't do this, and so commercial organizations can't trust and use the software for their business-critical work. I think what's going to be needed here if open source is to really take off in the cloud is for there to be these third-party organizations that identify useful open source software and deployment, maintain and support it on the cloud—making it available as a service with some sort of payment model.

**Yousif:** If you take a look at OpenStack, just as one example, I think it's getting a lot of traction in the industry and it's also proven itself as an element for standardization among service providers.

**Chase:** OpenStack seems to be a very big community and has managed to attract a lot of attention to its brand. We use OpenStack in ExoGENI. It's under the hood. I think that there's still a long way to go in terms of stability for these systems to really have big uptake as a basis for running mission-critical kinds of software. We hope that eventually they'll get there, but I think there's an important question here that we just touched on, which is it might be worth discussing the economics of open source. You know, these people aren't all necessarily working out of the goodness of their hearts. There has to be a sustainable economic model that drives the development of these systems for the same level of quality that we see in enterprise-grade commercial systems. I think the jury's still out on that.

**Grimshaw:** I was just saying it's not just the development, it's the maintenance, that's a real challenge.

For me, the most successful open source projects that have affected us on a large scale have ultimately been basically backed by the major corporations in one form or another by having their own teams work on it. In economics, there's a thing called the public good, and the problem with public goods is normally no one actor gets enough of the benefit for something to actually pay for it, but I think occasionally some of the big companies, whether it's IBM or Microsoft or Sun, formerly Sun, can reap that benefit, and so we have . . . the only things that will last for a long time are the ones where particular vendors can get enough of the benefit in the open source in order to recoup their money.

**Yousif:** Let's look at economic models and how you see them evolving to incentivize the use of cloud.

**Grimshaw:** This is a personal favorite of mine. I was an econ major as an undergrad, and there's been a lot of research on economic models from the perspective of the consumer, essentially the demand side of the function, but not as much on the supply side of the function, and in economics you always have to look at both sides. This gets back to something I said earlier—I think that this notion of buying and selling access to resources is important because, in the long term, there are always going to be organizations that want to have some of their own resources for their base load or for whatever reason, and they might be willing to sell that from time to time, and the question is how can they do it securely, obviously, but building models and systems that allow people to participate, not just as consumers, but also as providers of a resource is really interesting, and I think particularly in academia that might be interesting going forward.

**Fox:** I already commented that I thought this is a difficult subject because of the indirect way computing is funded via NSF projects like XSEDE [Extreme Science and Engineering Discovery Environment] or just on your campus. Computing's provided as a service, which is, as at our Indiana University, free to users. I expect this to change and, for example, campuses to focus on research computing support, not on research computers. The latter are likely to be cheaper in clouds.

**Grimshaw:** At [the University of] Virginia and at other institutions, people are already starting to charge for storage because they can't just keep giving people endless storage.

**Fox:** I strongly support that. I had an argument in some project I was on in this area. People insisted they couldn't use clouds because their local storage was free and commercial clouds charged for storage. I noted unsuccessfully that this was a flaky argument as universities (overhead) do pay for storage.

**Grimshaw:** But it's not free. At the University of Virginia, storage is not free anymore beyond some trivial amount, and the same thing is changing to cycles. We're moving toward a model in which people will have to pay for their cycles, and they may pay for them with an allocation. For example, the School of Arts and Sciences might put in money, which it got from overhead, as you said, to buy essentially a certain number of CPU hours and to give these allocations to the researchers internally as part of the overhead distribution process. I think we're going to see on campuses a realization that we can't give away computing power, storage, and all that kind of stuff for free forever because overhead, as a means of recovering those costs, just can't do it for the really big researchers, and we need some mechanism for people to be able to put stuff on grants. This is a very academic-focused notion of things, but that's the world I live in right now, is how do we provide large amounts of resources to new hires or big hires that we're trying to get and how do we build up the infrastructure because we can't give away cycles at the University of Virginia anymore. The only things we can give away are the things that we buy. Either we buy them with a machine in our machine room or we buy them from Amazon or Penguin or one of those guys.

**Yousif:** Okay. That's satisfying. One more question now. I'm sure you all heard about software-defined this or that like software-defined networking, software-defined storage, and software-defined datacenters. How do you see these going forward and their impact on cloud architectures?

**Chase:** I think it's very important. One of the things we're seeing, one of the ideas behind the ExoGENI project, is that the infrastructure as-a-service model is increasingly pushing out into the network. And so SDN [software-defined networking] gives tenants some ability to program the network. But there's another part of that, which is we're increasingly seeing layer-two circuit providers that allow you to allocate virtual network pipes from one place to another,

# OUR PANELISTS

**Jeffrey S. Chase** is a professor of computer science at Duke University. He's the senior faculty member in the systems area at Duke, and also leads efforts in scalable computing or the Duke campus. He has served as an architect in the US National Science Foundation's Global Environment for Network Innovations (GENI) project, and is working with collaborators at RENCI to deploy ExoGENI, a multicampus networked cloud testbed based on the Open Resource Control Architecture (ORCA) orchestration software developed in his research group. Chase received his PhD in computer science from the University of Washington.

**Renato J. Figueiredo** is an associate professor in the Department of Electrical and Computer Engineering at the University of Florida. His research interests are in the areas of virtualization, distributed systems, overlay networks, computer architecture, and operating systems, and he has cochaired the technical programs of the ICAC-2010 and HPDC-2013 conferences. His research has contributed to systems that were among the first to apply resource virtualization techniques in the context of grid and cloud computing, including In-VIGO, IPOP, SocialVPN, and Grid Appliance. Figueiredo received his PhD in electrical and computer engineering from Purdue University.

**Geoffrey Charles Fox** is a distinguished professor of informatics and computing and physics at Indiana University, where he is director of the Community Grids Laboratory and associate dean for research and director of the Data Science program at the School of Informatics and Computing. His research interests involve applying computer science to bioinformatics, sensor clouds, earthquake and ice sheet science, and particle physics. He is principal investigator of FutureGrid, a facility to enable development of new approaches to computing including clouds, grids, and distributed systems.

Fox received a PhD in theoretical physics from Cambridge University. He is a Fellow of APS and ACM.

**Andrew Grimshaw** is a professor of computer science at the University of Virginia. He is a cofounder and former chairman and chief technical officer of Avaki Corporation, and the founding director of the University of Virginia Alliance for Computational Science and Engineering (UVACSE). Grimshaw is the president of the Open Grid Forum (OGF), having served both as a member of the OGF's Board of Directors and as Architecture Area Director. His current projects are Genesis II, an open source, standards-based, grid system that focuses on making grids easy-to-use and accessible to non-computer scientists; and XSEDE (Extreme Science and Engineering Discovery Environment), the US National Science Foundation follow-on to the TeraGrid project. Grimshaw received his PhD from the University of Illinois at Urbana-Champaign.

**Paul Watson** is a professor of computer science and director of the Digital Institute at Newcastle University, UK. He also directs the UK's Social Inclusion through the Digital Economy Hub. In the 1980s, as a lecturer at Manchester University, he was a designer of the Alvey Flagship and Esprit EDS systems. His research interests include scalable information management with a current focus on cloud computing. Watson received a PhD in parallel functional programming from Manchester University. He is a Chartered Engineer and a Fellow of the British Computer Society.

**Mazin Yousif** is the editor in chief of *IEEE Cloud Computing*. He's the chief technology officer and vice president of architecture for the Royal Dutch Shell Global account at T-Systems, International. Yousif has a PhD in computer engineering from Pennsylvania State University.

and that's an important part of the cloud integration and interoperability and federation story—the ability of tenants to create private networks that span multiple cloud sites, and then to manage those networks. There are a lot of benefits in terms of performance and quality of service and possibly data privacy as well, and to the extent that there's uptake of that idea, the next question is how do you manage interconnection of these private networks with each other to allow data sharing among tenants in a controlled way, and also to connect other resources in the campus or enterprise into these cloud-allocated networks.

**Grimshaw:** I'm going to say something controversial. I mean, this is one of the . . . so I think you're more of a networking guy and I'm a higher-level abstractions guy. One of the biggest challenges that I find, for instance, recently doing some work here in Germany, is that people try to define security at the networking layer, and I'm not saying that you're guilty of this, when it really should be done at a layer where you actually know who the principals are and what it is they're trying to accomplish. And so, to some extent, I'm against trying to push more functionality into the network because then the networking guys basically take over control of that and block the ability to do it at the higher layers of the stack. So if I were to say one thing to networkers as somebody who tries to build out these kinds of heterogeneous infrastructures that couple multiple organizations together is please stay out of the authorization business.

**Watson:** For me, I think it's interesting because we are moving to this world where there are lots of useful services that can be combined to build applications. However, organizations will want to build applications that meet particular requirements in terms of scalability, performance, and dependability. At the moment, it's largely down to developers to achieve that, and most organizations don't have the skills or the resources to do it. Therefore, a research challenge is to move to a world in which there are high-level ways to describe the nonfunctional properties that we require of our applications that are built by combining these cloud-based services together, and then have a tool that will generate that application for us.

**Figueiredo:** I want to add on the software-defined networking side, that I think I agree with Jeff that it's very important to be able to create virtual networks across multiple providers. I'm not sure how long it's going to be before deployments of low-level software-defined networking approaches will allow interoperability, but one aspect is you need to have some guarantees from the network side, and it could be—even if it's not security, just from the management side, being able to have the ability to manage your own IP addresses and to manage them in a way that you're able to move freely around multiple providers is important. I really want to deploy services across multiple clouds, so in one of the projects I'm involved with (IP-over-P2P, IPOP), we're looking at, you know, pushing that software definition all the way up to the user and enabling end-to-end networking that doesn't make any assumption about the controllability of the network infrastructure. I think there might be a convergence moving forward between that style of virtual networking (user-defined) and software-defined infrastructures.

**Fox:** I've already mentioned I think this concept could be important and it's key to actually be able to define what you did when computing and be able to do reproducible computing, which is certainly one thing you need to be able to do in the future. Software-defined systems are also key to interoperability between heterogeneous computing environments.

**Yousif:** We have 10 more minutes left for the roundtable. I would like each to take two minutes on final thoughts on what we discussed and when do you see them happening?

**Chase:** The technological and commercial landscape will continue to be dynamic. The markets will continue to fluctuate. I'm sorry. I'm sort of joking. I think the main point is that we're in a time of great change, and that's certainly going to continue. I think we've touched on a lot of the key topics.

**Grimshaw:** I'm going to observe that if you think about cloud or on-demand services, you know, it's something that people have been thinking would happen for a long time. It has happened now, and it started at infrastructure as a service at the lower layer, and it's clearly caught on and for the first time, people are seriously outsourcing part of their computing, and I think that's a really good sign. I think, though, that the interesting challenges and opportunities are in combining higher-level services, which may or may not be hosted on IaaS clouds. So I think, then, in a lot of ways, IaaS clouds and Amazon have broken sort of this conceptual and perceptual issue with the community about outsourcing their stuff and made it acceptable. The higher-level services will be the next thing to come along, and that's

where the real benefits will be down the timeline. That's my belief going forward for the next, I'd say 10 or 15 years, is that we'll see an explosion of those higher-level services and interoperable services, and the challenge is how to make them interoperable and work together.

**Figueiredo:** If you look at Amazon, it has really broken ground in this idea and made it possible for people to think in the context of this paradigm of computing as a service at a fine granularity and with elasticity, and so it takes time for some of the higher-level services to begin to use that capability in a more meaningful way, and for others to pick up and be able to compete with the providers. I wonder if, in the future we're going to see the ability for smaller-scale providers to be able to differentiate and have the same style of provisioning that you have now from large providers, bringing more diversity and, as Andrew pointed out earlier, the ability to use and contribute resources to a larger pool of available computing as a service.

**Fox:** So I guess I sort of mentioned this already. I think cloud computing's a pretty disruptive technology. It will impact a lot of the way we do things and it will change a lot of the organizations we have, and we have only just seen it start. It will get much more dramatic in the future.

**Watson:** I think that cloud computing has become really important for research, giving bright researchers the opportunity to acquire the resources they need to do their experiments quickly, and for other organizations because of the agility that it brings them. In particular, start-ups can now be launched from a founder's spare bedroom with only a laptop and access to a cloud, whereas in the old days, they might have had to trawl around trying to get VC [venture capital] backing so as to be able to buy lots of servers that would sit relatively idle in their machine room waiting in case the service became a success. So, I think it's already had a hugely beneficial effect. Also, as people have said, I think if we move to high-level services in the cloud, this is going to allow more organizations to benefit. For this to happen requires two things. One, software vendors need to move to producing the scalable, cloud-based services from which organization can build their applications. Two, we need high-level tools that allow users to combine these services together to meet both their functional goals and their nonfunctional requirements, such as security, scalability, and dependability.

If I had to predict another thing for the future, I expect private clouds will dwindle away as confi-

dence in public clouds increases. So I think that by 2020, private clouds might be almost gone. As Geoffrey says, clouds are a disruptive technology, but I think it's also worth thinking about whether anything can disrupt this disruptive technology. For example, what would be the analogies to the financial crash of a few years ago when most experts thought that the current financial system would continue quite happily for many years, but it didn't. I don't think this is likely, but I think it's worth considering what would happen if one big public cloud provider failed or withdrew from the cloud market, I think that this could have enormous implications. As I said, I think that's unlikely, but I do think that anybody using the cloud should at least consider what they would do if that happened.

**Chase:** I can throw in one closing thought if we have time for it.

I was just going to say that we may very well see new kinds of applications enabled by lots of computational power in the cloud that's easily accessible on the network. We already see some of the applications for speech recognition on mobile devices using back-end cloud processing, and increasingly we may find that the clouds are more tightly integrated with applications that control infrastructure, you know, smart cities, those kinds of things because of the large amount of compute power on tap and the elasticity that the cloud offers.

**Yousif:** Well, thank you all. I agree with the last comments about cloud computing being a paradigm, and we're going to see further explosion of high-level services. We're likely going to see much fewer infrastructure service providers, mainly because you need a lot of capital for it. We have issues regarding security and privacy. That's very likely going to be a good area of research.

Open source will likely play a bigger role going forward, and lastly, standardization will need to be pushed further for the market to adopt clouds more heavily. ●●●

### References
1. F. Manjoo, "Larry Page on Google's Many Arms," *New York Times*, 25 June 2014; www.nytimes .com/2014/06/26/technology/personaltech/ a-reach-too-far-by-google.html.