Tactical Cloudlets

featuring Grace Lewis as interviewed by Suzanne Miller

Suzanne Miller: Welcome to SEI podcast series, a production of the Carnegie Mellon University Software Engineering Institute. The SEI is a federally funded research and development center sponsored by the U.S. Department of Defense and operated by Carnegie Mellon University. A transcript of today's podcast is posted on the SEI website at sei.cmu.edu/podcasts.

My name is Suzanne Miller. I am a principal researcher here at the SEI. Today, I am pleased to introduce you once again to Grace Lewis who has spoken with us before. She is the deputy lead for the SEI Advanced Mobile Systems Initiative and technical lead for the Edge-Enabled Tactical Systems research team. In today's podcast we will be discussing her research on tactical cloudlets.

First, a little bit about Grace. Grace's main research interests are mobile computing, service-oriented architecture, and cloud computing. She has more than 25 years of professional software development experience in both industry and research environments and is also active in mentoring student teams in Carnegie Mellon University's Master of Software Engineering Program. Grace is a frequent guest on the podcast series where she gives us the latest on her cloud and mobile computing research. I have had the privilege of working with Grace in the past, and I am particularly glad to have a chance to catch up on her current research today. Welcome, Grace.

Grace Lewis: Thanks, Suzy.

Suzanne: We have interviewed you previously to talk about your research to help soldiers who use smartphones in battle. At the SEI, we tend to refer to that as on the tactical edge. It also refers to first responders and other people that have to be on the edge of where there's connectivity. For those listeners who haven't heard this term previously, I'd like to start off by explaining the challenges that soldiers and first responders face in this type of environment.

Grace: Sure. First of all, when we talk about tactical edge, by "edge" we really mean at the edge of the connected network infrastructure. Soldiers and first responders who operate at this tactical edge face challenges of

- first of all, dynamic missions because there are lots things happening, and things can just change in this type of environment;
- limited and intermittent network connectivity because, as I just explained, it is at the edge of the connected network;
- limited computing resources because, as Suzy said, we are envisioning people using smartphones, which have much less computing resources than a regular desktop or connected server; and finally,
- high levels of stress because the conditions under which these people operate, which could be disaster environments, are very stressful.

So, if you combine all these conditions, it makes operating in this environment very challenging, and that is the area that we work in.

Suzanne: So, you are trying to extend the capabilities of mobile devices used by these folks with these multiple limited resources that you are dealing with. You have got a concept of cloudlet-based cyber foraging. Let's talk about what cyber-foraging is and how cloudlets relate to that.

Grace: OK, so, cyber foraging is an area of research and work within mobile cloud computing that leverages external resources, whether these are cloud servers or local servers that we also call surrogates, to augment the computation and storage capabilities of these resource-limited mobile devices, which could be smartphones, while at the same time extending their battery life.

So, there are two main forms of cyber-foraging; one is computation offload, where the idea is that you would offload expensive computation that is supposed to run on the mobile device; you offload it to a server to extend the battery life and increase the computational capability because it is using more powerful cloud resources than the ones on the mobile device.

The second form of cyber-foraging is called data staging. The idea is to improve data transfers between mobile devices and the cloud, and vice versa, by temporarily staging data and transit on these surrogates, which we call cloudlets.

Suzanne: For people that don't know, a lot of the data transfer is almost like what we call record-by-record or even field-by-field. There are lots of instances of data moving. So, what you are really talking about with staging is grouping packets of data together, so that you are not always transmitting, you are actually limiting the amount of transmission.

Grace: Correct. Actually, there are multiple forms of data staging. What you talked about is, what we call them, is outbound data processing and inbound data processing. In one case, you are gathering data on a mobile device, and you are trying to send it to the cloud, which may or may not be available, right?

Suzanne: Right.

Grace: The other one is inbound processing where you have data in the cloud, and you want to get it to the mobile device, but you don't want the mobile device to get all the data. What you do is, for example, you place a cloudlet—which I have yet to talk about—and you place it in the middle, and that cloudlet does a little bit of the processing, so that the mobile device doesn't get overwhelmed.

You can also talk about pre-fetching, which is another form of data staging, where what you have is the cloudlet—because it is a computational unit—what it is trying to do is it is trying to guess what data the mobile device needs. Then it does some pre-fetching. So, for example, if you are in a certain region, if you know that a mobile device is in a certain region, you might pre-fetch data particular to that region because you have the feeling that that mobile device is going to be able to use that data.

So, there are different forms of data staging, but in general, that's the idea: to be able to concentrate data in this intermediate surrogate, which we call a tactical cloudlet, so that it more efficiently gets used by the mobile device.

Suzanne: So, talk explicitly about what is a cloudlet and how does it help with this staging and offload of computation.

Grace: Sure. The easiest way to think about a cloudlet is really as a discoverable data center in a box, the smallest private cloud you can think of. The essence of tactical cloudlets—in addition to being discoverable— is that they are in single-hop proximity of the mobile devices that use them.

What I mean by discoverable is that a mobile device can basically discover cloudlets around it because what the cloudlets do is they broadcast their presence. So, a mobile device can say, *OK, I have two or three cloudlets around me, and I am going to be able to use them in this way or another*. They are in single-hop proximity meaning that I am talking about Wi-Fi as opposed to 3G. That is key because not only do Wi-Fi connections have greater bandwidth, but also there is a lot of evidence that shows that Wi-Fi, for example, consumes less battery than 3G communications.

In addition, tactical cloudlets are virtual-machine-based. What this allows us is to take advantage of what big data centers do for management, for scalability, for elasticity, except that we do it at the edge.

Another advantage of a VM-based solution, or virtual-machine-based solution, is that really you can package any legacy capability inside of a virtual machine. Basically what you need to do is build a very thin client for the mobile device to use that capability.

So, if I were to summarize, *What is a tactical cloudlet?* I would say it is a forward-deployed, discoverable, virtual-machine-based server that can be hosted on vehicles or other platforms to provide infrastructure to offload computation; to forward data, stage data for a mission; to perform data filtering to remove unnecessary data streams, for example, intended for dismounted users; and finally, to serve as collection points for data heading to enterprise repositories.

Suzanne: This is a really important concept, and there has got to be lots of interest in this. You have spoken at some conferences about this, but you have also come up with five approaches to provisioning these cloudlets. So, if this is data center in a box, there have to be different ways of accessing that and different ways of configuring those.

So, your research has developed these five approaches. In the past, we have interviewed you about optimized virtual machine [VM] synthesis and application virtualization. Give us a brief recap of those, as well as the other three approaches you have discovered.

Grace: OK, so let me provide some background, especially for those listeners that have not heard the previous podcasts. One key thing—in addition to being discoverable and virtual-machine-based and all that—is that cloudlets are intended to ideally be used disconnected; disconnected from the enterprise because again there is no guarantee about the connectivity that you are going to have between the cloudlets and data centers or the enterprise. A key concept behind this is something called cloudlet provisioning, which is, *How do you get the computation or the capabilities on that data center in a box or that data on that box so that people can use it in the field?* We call this cloudlet provisioning mechanisms.

Our first piece of work looked at something called optimized VM synthesis. This is because when cloudlets were envisioned, there was a process by which they were provisioned called VM synthesis. This is something that was developed by our collaborator on campus. His name is Mahadev Satyanarayanan or he goes also by “Satya.” What VM synthesis is all about is the following: what gets transferred between the mobile device at the cloudlet at runtime is something called an application overlay. This is a case where we are doing cloudlet provisioning at runtime. So, the way an application overlay gets created is that, in advance, what you do is you start up a base VM. A base VM is whatever you defined as your baseline. Then what you do is you install an application inside this base VM. Once it is installed, you suspend it, right? So, once you suspend it, then you have two different basically binary files because that is what they end up being. You have a file that represents the state after everything was installed, and you have a file that represents or a set of files that represents the state of that base VM, what your baseline is. So, because these are binary files, you can calculate the binary difference between these two. This basically constitutes an application overlay. So, at runtime, in the original VM synthesis, that mobile device would transfer this application overlay to the cloudlet at runtime. Then, the cloudlet would apply that overlay to the same base VM from which it was created to constitute, we call it a complete VM.

We made some optimizations to this original VM synthesis process that was proposed by Satya because the application overlays, at least as they were originally envisioned, were extremely large. We wanted to see if we could make these application overlays smaller.

Suzanne: That goes against some of the idea of the tactical edge.

Grace: Exactly. I want to go back a little bit too because when cloudlets were envisioned, they were not meant to operate in these types of environments; they were more intended towards the more regular type of everyday scenarios. So, we did some optimizations. We also wanted to speed up the process itself because, not only was it taking time to transfer the application overlay, which is quite large, but it was also taking time to do the synthesis process of applying the overlay on top of the base VM. Even with all these optimizations, it still was really large to handle in edge environments. We did some experiments, which I will talk about later, but the overlays were between 55 and 300 megabytes depending on the type of application. That is really hard to handle in tactical edge environments.

Suzanne: Sure.

Grace: What we started doing was just looking at different approaches but still trying to provision from the mobile device itself. So, we looked into something called application virtualization. The idea behind application virtualization is to create these virtualized applications or packages where you try to wrap or contain everything that the application is going to need inside a package, so that basically you are isolating it from the operating system. So, basically it can work self-contained independently. There are lots of tools out there that you can use. In particular, we used two tools: one called CDE [code, data, and environment] for Linux-based environments, one called Cameyo for Windows- based environments.

What happens when you use these tools in these environments for application virtualization is that a runtime component intercepts all system calls from the application and redirects those to resources inside the virtualized applications. So that way you don't have any dependencies with the specific operating system or any differences that could be between versions and builds and things like that. Basically the virtualized application has all that it needs, and it doesn't rely on specifics of the operating system.

The advantage of application virtualization is that what is transferred at runtime from the mobile device to the cloudlet is the virtualized application package, which is a lot smaller than an application overlay.

Our experiments created packages between 15 and 70 megabytes, which is much smaller than the numbers I mentioned before, but if you think about it they are still too large for edge environments. In addition—something that we have found and that a lot of people [have found] that have used these tools to package applications—is that it is very hard to guarantee that all the dependencies are captured. This could lead to errors at runtime.

Suzanne: Not a good thing…

Grace: No.

Suzanne: …when you don't have enterprise resources to make corrections.

Grace: Correct. In addition to that is another challenge that I didn't mention. If you think about where these systems are deployed, you don't really have IT people.

Suzanne: Typically not.

Grace: So, it is very hard to fix any problems with that.

Suzanne: Yes. OK. What came next?

Grace: Okay, so this made us take a step back and say, *In these environments, is it really that important to provision from the mobile device? Why don't we look at what data centers do, what cloud providers do?* For example, if you are a user of a cloud infrastructure, you are not provisioning the cloud at runtime with your computation. It is already there, right?

So, what we did is [ask ourselves], *OK, what if, instead of trying to provision from the mobile device, we try to just leverage what the data centers do?* So, we created a third cloudlet provisioning mechanism called cached VM. Basically what we are doing here is that we package each capability in something that we call a service VM. You can think of service in the same way of like a SOA [service-oriented architecture], a service-based type of system. It is a single capability. It is self-contained. In our case what it exposes is basically a service VM ID.

What happens in this case is that you would pre-provision the cloudlet with a lot of these service VMs that correspond to mission-specific capabilities. In that case, basically when you're going to go out on a mission, you are trying to think, *OK based on the mission that this person is going to execute, what would be the capabilities that would be necessary? Does this type of mission require speech recognition? Does it require natural language translation? Is it going to require this type of data, map data, etc.?* So, you provision these cloudlets with a lot of these service VMs that have, like I said, a unique identifier. At runtime what the mobile device does is instead of saying, *Please deploy this capability for me*, it says, *Do you have this capability?* So, it says do you have the capability that matches this service ID?

Suzanne: On the cloudlet.

Grace: Correct. If it does, then it starts a service VM. The advantage obviously is at runtime, I am not transferring a big application overlay or a packaged application or anything; it's just a service ID. The disadvantage is that the cloudlet has to be pre-provisioned. So, for example, if I don't have that capability on the cloudlet, it is a problem. At this point, I have two options: either *Well, I just don't have the capability* or, if by any chance that cloudlet is connected to the enterprise at that time, the cloudlet could say, *Does the enterprise have this capability and bring it over?*

Suzanne: You don't know if that is the case, but that is a possibility.

Grace: Oh, absolutely.

Suzanne: In provisioning the cloudlet you would look at historical data for these kinds of missions, provision the highest probability ones. You would have to balance because you've got a space limitation in the cloudlet as well.

Grace: Correct.

Suzanne: *What are the things that are not as likely to be used?* You would make those decisions at the enterprise level.

Grace: Correct. Also, if you think about how the client/server model works with these cloudlets, the idea is that what runs on the mobile device is a very thin client. Then you have all the computation-intensive stuff running on the cloudlet. So, with this particular provisioning mechanism for Cached VM, the mobile device would have to have the client. Basically, I am running this app, and say *Do you have the matching server for this?*

So, I run the client, *Do you have the matching server?* In our fourth mechanism, we said *OK, let's take it even one step further*. Suppose a mobile device doesn't have anything, it just knows that *I need some kind of face recognition capability. I need some kind of speech recognition capability*. So, we implemented something, cloudlet push, which is basically an app store.

I said before, at runtime, instead of saying *do you have this particular service ID*, it basically says *do you have this type of a capability?* What happens at runtime are two things in parallel: One is that the client gets sent from the cloudlet to the mobile device, which is why we call it cloudlet push. It gets installed on the mobile device. At the same time, we start the backend server for it. While this application gets installed on a mobile device, the service VM gets up and running. Then, at the end of this whole process, you are back to the previous case of the Cached VM basically.

The advantage again is a small payload. Because the client is so tiny, what gets sent from the cloudlet to the mobile device is very small. So, you also have that advantage of the small payload. The disadvantage now is that the cloudlet would have to have as many versions of that client for the different types of…

Suzanne: All the different platforms.

Grace: Platforms.

Suzanne: Sure.

Grace: …which could or could not be a problem. If you have a standardized client, then it is not a problem. If you have an open [environment], you can have whatever client you want.

Suzanne: A bring-your-own-device kind of environment, a little more challenging.

Grace: Then it becomes a little bit more challenging and so we were pretty happy with Cloudlet Push, but then we said *OK, what do data centers do?*

We came up with a fifth cloudlet provisioning mechanism, and we call it on-demand VM provisioning. The way servers and the cloud get provisioned in industry or in the enterprise is using cloud provisioning tools. An example of this type of tool is something called Puppet. There is also Chef.

Basically what these tools do is, if you are the administrator for a bunch of servers in the cloud, you run scripts in these tools. So, for example, if a customer, in this case which is your organization, says, *Oh, I want a VM with this, and this, and this in it*, then you would say, *OK, you want this type of database, you want that type of different tools in it.*

What it does is it creates on the fly a VM or a space that matches whatever you want, right? So [we asked ourselves] *What if, what if we did that at the edge?* which was a little bit challenging and a little bit out there. We wanted to do it anyway, which is *what if at runtime what I send from the mobile device to the cloudlet is one of these scripts?* Basically, I am not asking for a specific capability. I am not asking for a particular app or anything like that. Basically I am saying, *Please construct this service VM for me*. So, at runtime the tool runs on the cloudlet in the same way that it would run on the cloud. It starts putting pieces together. It starts a base VM and installs a database in it. It installs a face capability, it installs other stuff in it. At the end of the process, then you have a service VM.

Now, the advantage of this is extreme flexibility because basically, as long as you have the pieces together, you can put together whatever VM you want. That advantage is also a disadvantage because now you have to be certain that all those pieces are on the cloudlet or that maybe you are connected to the enterprise and those pieces are available somewhere. In practice, what also happened is it ended up being a very lengthy process, because…

Suzanne: That is what I was wondering about, the time to construct.

Grace: You are starting to put pieces together, but we wanted to see if this was possible at the edge. Is it possible? Yes. Could it be a potential way of doing it at the edge? Yes, but you have to understand the disadvantages that come with it.

Suzanne: OK. So, you are basically going from packaged to piece parts…

Grace: Correct.

Suzanne: …and construction. So, you've got a spectrum of ways, which could apply in different settings.

Grace: Absolutely.

Suzanne: I think you have reached the boundaries on that one, but then you also conducted some experiments in addition to just figuring these out.

Grace: Sure.

Suzanne: What is more effective for people at the edge in different settings? So, what were some of the results of those experiments and how did you conduct those?

Grace: OK, sure. We compared the five cloudlet provisioning mechanisms in terms of—from a quantitative prospective—payload size, energy consumption, and also application-ready-time, which we define as the moment from which I say *I need this capability* until when the cloudlet says the capability is ready.

In addition to that, we did a qualitative comparison because, again it is not just about the numbers. We started to look at advantages and disadvantages of all the methods in terms of how easy it is for a user, how easy it is to deliver.

Suzanne: Cognitive load is an issue for people that are in risky situations.

Grace: Correct. All these things. Basically, combining the quantitative numbers from the qualitative, we decided to establish a baseline. Basically our tactical cloudlet implementation baseline is a combination of cached VM and cloudlet push.

So, it's a combination of, you can use both. If I don't have any client on me, I can say *OK, do you have this capability?* Once I have the client, then I go back to cached VM. The application gets only installed once.

We call it a combination of cached VM and cloudlet push. It really enables lower energy consumption on the mobile device. We also think it places less requirements on the mobile devices themselves. Finally, it really simplifies the provisioning and tactical cloudlet environments because, when you are pre-provisioning your cloudlet, you are placing all the capabilities in there including their clients.

Now you have everything. You really have the data center in a box, but you also have the app store in a box. We think it's the combination, the best of both worlds.

An additional advantage of combining both mechanisms—and I already talked a little bit about this—is that if the mobile device already had the client app, it can simply just look for a matching service VM. If not, it can obtain it from the cloudlet. So, it's doing the app store.

The tradeoff, and I also talked about this before, is that it relies on cloudlets that are pre-provisioned with server capabilities that might be needed for a particular mission, or that the cloudlet is connected to the enterprise, even if just at deployment time to obtain the capabilities. We would argue that this requirement is not unreasonable in tactical-edge environments, and that it makes cloudlet deployment in the field easier and faster while leveraging the state of the art and the best practices from the cloud computing industry. Our pre-provisioned VM-based solution also promotes resiliency and survivability by supporting rapid, for example, live VM migration in case of cloudlet mobility, in case you discover more powerful or less loaded cloudlets, or if there is some kind of an unavailability due to disconnection or disruption. It also supports scalability and elasticity because you can start and stop VMs as needed based on the number of active users, which is what is done in industry. What is interesting again about edge environments is that user size is typically bounded. You know what the size is, when you deploy a mission you know how big the group is going to be. So, we're not talking about what I call the Starbucks scenario where you can have a lot of people coming in.

Suzanne: It's unbounded.

Grace: Correct, it's more bounded, so it really lends itself to these types of environments.

In addition, if you think about a lot of the applications that are necessary in the field, they are very request-response. *I took a picture; please process this image for me. I grabbed this voice here; can you please process this. I grabbed a sample of something.* So, it's very request-response. That also lends itself a lot to these cloudlets and also to disconnected operations where you are not constantly trying to gather data, but you are sending a request that is being…

Suzanne: So, you are not doing sensor fusion or data fusion.

Grace: No, you are not doing anything like that.

Grace: You could.

Suzanne: That's a different interview.

Grace: Yes.

Suzanne: You are trying to get an answer to a question. Again, it is back to that, *We are in very pressure-cooker situations, and we are trying to limit what we need just to get to the next thing that we need to do typically*.

Grace: Correct. So, because of this request-response type of scenario, you can imagine a cloudlet-based solution in which you send a request, right? You send something for processing, and all of a sudden you lose conductivity, well a cloudlet could store that, for example, for you until…

Suzanne: Right.

Grace: You get reconnected and say.

Suzanne: *Send.*

Grace: Correct. Or, for example, it could find another a cloudlet and say, *Hey, maybe there's another cloudlet that could get this information to you.*

Suzanne: So one of the possibilities of this is communication cloudlet-to-cloudlet.

Grace: Absolutely.

Suzanne: We don't want to forget that as one of the resources that could amplify this sort-of smartphone, edge environment.

OK. You didn't do this by yourself.

Grace: No.

Suzanne: I know that you have got a wonderful team of researchers…

Grace: Absolutely.

Suzanne: …that you work with in this area to help the warfighters to help first responders make better use of their limited resources in the field. We've talked to Jeff Boleng and Anind Dey; they talked about group-context-aware computing as a way to improve situational awareness. We've talked to Soumya Simanta, who talked about situational awareness mashups. That has led to a whole new area of research within your group called edge analytics. So, what else have you guys got going on with you and members of your teams that people can look forward to?

Grace: A lot.

Suzanne: It is very bad for Grace to be bored. I just want to say that.

Grace: I'll talk first about tactical cloudlets because that is the work I'm closest to. The area that we are moving into is the area of *trust*. In particular, establishing trusted identities in disconnected environments.

Suzanne: This is part of *discoverable-but-trusted* is what you are trying to add into that kind of equation?

Grace: Correct. So, our current tactical cloudlet implementation, the baseline that I talked about, has basic security features, and, for the most part, relies on the security provided at the network level. We realize that this might be enough in some scenarios, but it's definitely not enough in others. What we're looking at is ways of establishing trusted identities between mobile devices and cloudlets that don't require reach-back to an online trusted authority, or, for example, rely on certificates that might be expired because in the field there might not be a way to validate.

So, we are working with our CERT colleagues to establish some kind of trusted identity for these cloudlets, so people will feel more comfortable with the trust of the solution that is implemented.

Jeff Boleng and his team in collaboration like you said with Dr. Anind Dey from campus. Jeff leads a task called Information Superiority to the Edge. The goal of this task is to leverage not only individual context, but also the context of the people that are deployed as a group to make decisions with respect to, for example, resource optimization or data visualization that benefit the whole group and not just the individual. You know, it is like the sum is greater that the parts.

This past year they focused on activity recognition and being able to display the right information at the right time to the right person based on, not just what the person is doing, but also what the group as a whole is doing. So, it is really about activity recognition. The data is still being processed, but the results are looking very promising to be able to detect what a group is doing.

Soumya Simanta and his team, they lead a task called edge-analytics. This task focuses on developing system architectures for analyzing large streams of data at the edge. Their proposed architectures and capabilities leverage what we call fidelity-timeliness tradeoffs; meaning that the system might not be able to provide a 100 percent precise answer in human real-time with the resources that it has at the edge, but it might be able to provide an 80 percent answer in human real-time. We talk about seconds.

The team has been very successful in using social media streams, such as Twitter, as an example of the analysis capabilities, as well as the visualizations for the data streams. For example, we can do sentiment analysis. *Are these happy tweets or sad tweets?* We can do named entities; meaning, *What is tweeted about? Is it an organization? Is it a person? What are the trends showing?* [Also] topic modeling to be able to bucket, for example, tweets into topics. We are using Twitter simply because it is an example of a large data stream. You can imagine using, like you said, data coming from sensors.

But, basically it is any type of large data stream. However, this year what they are focusing on is assigning confidence values because anybody can tweet. Anybody can tweet about anything.

Suzanne: Sure.

Grace: There have been many, many scandals and many things recently about...

Suzanne: Repetition creates truth.

Grace: Correct. What Soumya and his team want to work on this year— because we've had a lot of success with social media streams—is assigning confidence values. Not only that, but showing what was the chain of reasoning because, again, we're at the edge. We can't focus on the 100 percent solution. If you tell someone is trying to visualize all this information saying, *Hey, I think this is true, and I'm 80 percent confident. This is what, I, meaning the system, did do to say, this is the chain of reasoning, why we think it's 80 percent true.*

Suzanne: If you have more information about context than the system does, you can evaluate that, see that that element of context was not taken into effect when they were doing the analysis and make your judgment a little bit more concrete.

Grace: Exactly. So, instead of trying to provide like an automated, *I already did all this analysis and I'm 100 percent confident.* It is really up to the user to say, *OK, based on what I'm seeing, how should I react to this data?*

Suzanne: OK.

Grace: Finally, we are kicking off a brand new task in precisely sensor fusion, which is *Can I integrate data that is coming from these social media streams*? It is being used a lot in a lot of these environments that we work in. *Can we combine it with data coming from real sensors to be able to provide an even better picture of situational awareness at the edge?*

Suzanne: Cool. So, you're not bored I can tell.

Grace: I am not.

Suzanne: And, that is a good thing.

You are keeping everybody busy with all of these streams of research, so we will have future discussions about some of this stuff. I am particularly interested in the sensor fusion especially, I mean the way smartphones have incorporated sensors just for everyday use, some things that have to be amazing are possible here. So, that is very exciting. Thank you for telling us about your cloudlet work today. I think the provisioning things and the idea of a data center in a box is going to be very important both for the warfighters, as well as for first responders in a lot of different areas. So, I think that's very important work.

If you, our listeners would like to view Grace's paper related to this, as well as all of the SEI technical reports and notes, please visit resources.sei.cmu.edu. Click on the author tab and then on Grace's name in the author's A to Z index.

Grace has also authored several blog posts on the SEI website at blog.sei.cmu.edu, click on the cloud computing tag or on Grace's name under the author index.

For another information resource on the research Grace's team is doing in pervasive mobile computing please see their *Our Work* website at sei.cmu.edu/mobilecomputing/research/index.cfm.

This podcast is available on the SEI website at and on Carnegie Mellon University's iTunes U site. As always, if you have any questions, please don't hesitate to email us at info@sei.cmu.edu.