<meta name="viewport" content="width=device-width; initial-scale=1.0; "\>Event ID: 1933475
Event Started: 5/9/2012 6:00:00 PM

>>Good afternoon, everyone. Thank you for attending our mobile platform webinar development series. We have had the windows platform and the blackberry platform present and today we have a specialist on the apple iOS plat form. Our speaker today is Tim Hoechst. He's from the Agilex company and he's developed apps for divisions of the government. Today, Tim's going to tell us about developing on the Apple iOS platform and he'll talk for about 45 minutes and then we'll open it up to questions and answers at the end. If you could please IM in the little chat box your question, we can ask it at the end. Tim, thank you so much for joining us today, take it away.

>>> Terrific, thanks so much for having us. What I'm going to do is spend a few minutes working through a brief agenda here. We're going to start talking about some high level app development issues, some of the characteristics we see in mobile apps, particularly on the Apple platform may or may not be true on all mobile platforms, but something we have seen in government. We're going to spend an extra moment focusing a little bit on security, building security into apps, which is the highest priority, probably, we talk -- to everybody we talk to in government but we're going to talk about the process one has to go through for building and deploying applications in the Apple environment. At the end I'm going to talk a little bit about an example of an app that we focus on and then we'll save time for Q and A at the end. Just to note, our primary focus today is on app development through Apple's commercial ecosystem rather than building apps in the enterprise. A lot of what we do in government is building an application that runs within a government network to access internal systems and if folks have questions about those things we can talk about it but today I was asked to focus on the process of writing applications that are more likely to be citizen based and twitted through the Apple iTunes store. I want to talk about app development, it's not primarily not what we're going to talk about. I was reminded of an old Steve martin joke, I can tell you how to make a million doll dollars and not pay taxes, first, make a million dollars. That's what we're going to talk about here, but I can't do that without talking about a few key issues associated with app development in government, so for us, the most interesting apps are one that have an information system behind them, not necessarily just stand alone apps but they're exposing enterprise data whether it's inside the enterprise or out on the internet. There are couple of things that we have seen in building these types of apps that I thought everybody might find useful. First, traditional software development methodologies are falling very much out of favor in application development across the board, but in mobile application development we have seen that agile development is popular because of the way that organizations want to see demonstrable results so keep in mind that if you or your organization is starting to think about acomed Agile systems, this is a good place too experiment with that. Before coming to Agilex, I spent 120 spent -- 20 years at oracle and lot of that was building client semper information systems. As we move to the web and we think about applications having their functionally centralized in the network, we changed a lot of the way we build those systems but with mobile development, you think of these mobile platforms really as client computers that are connecting over a network to a back end information source and they have a mobile oop -- operating system and they have mobile memory and mobile processing capability. What's different, really, between the way we built applications for mobile devices and the way we did it in the client semperserver area is that these apps are more homogeneous. For those of you who lived through building a lot of client server systems you knew some of the challenges were writing an application that would run no matter how people configured their desktop machines and they had all the freedom in the world but then also getting application software out to those devices was difficult and cumbersome and inscient. We in -- inconsistent. We have the same kind of power but we have relatively consistent information and it's manageable and we can beam our applications to it. When we think about how an app is architected, it's much more like a client server app and that's assuming we're building native apps. What we are going to spend most of our time talking about

today are native applications, written to run on the local device even if they access information remotely, but there are also mobile enterprise application programs which are tools that are used to assemble apps and have them either run inside a shell on a variety of different platforms or actual compile native apps, the idea that you run it once and it can return on many platforms. It is an architecture that you should remain aware of, especially if you have intentions of your apps apps running more than one platform. We can also have a web architecture where the app is not really on the device as much as running in the network and just the user interface is on the divide. HTML5 is giving us a way to write apps that look and smell very much like the native apps but the functionally that make the app what it is on the network. Those architectures are important but for this discussion, we're going to talk about native. I want to emphasize that unlike any other platform, the user experience is really important, most notely because the bar has been set very high by the hundreds of shows of applications that are out there. Folks expect with a tap or two or a swipe or a shake of the device to quickly get to what they need, so I encourage you not to just put your field and tab and buttons based enterprise applications to the mobile platform. That works maybe okay on an iPad but it doesn't really work on an iPhone and so think about how you narrow your function fallly -- functionality and things that have to be done at the moment rather than trying to build it on a user interface and cram it all on that screen even with an iPad. We had one customer in the federal government that has doing a field survey type of application and they sort of had somebody port the experience of the old survey tool, this was predominantly paper based to the iPad and, yeah, it worked, and yeah, you could navigate around but it was not exciting. When we took it and made each question itself own screen and the way you accepted your answers was with touches and swipes and if you turn the device sideways you got help about what the question meant, just totally different things, it changes their thoughts about it too. Like many of those architectures, we want to put the logic in the network when possible and bill services providing access to those systems, not put all the functionality in the client's device. This is one lesson we learned in client server that when we put too much in the client app, we have to employ a new app to change it. It also means that we make an app on Android and iOS and others. We have to think a lot about dis disconnectivity in our apps. One we built was for Amtrak where the conductors needed access to their information even in a bridge and the importance of this was based on the use case but it's something to consider and something we didn't have to consider in the client server era because desktop machines did not become connected and this last bullet, the kitchen sink, I encourage you not to cram every feature, including the kitchen sink, into your app. We used to want to make sure it could do everything it could do in that first version and our experience has been that it's much more helpful if we pick some focus functions. One thing we here from everybody is they would like mobile applications to do their time and attendance. They want their folks to be able to enter the time they work but we don't have to build an entire HR system employed too a mobile device. We need to build a very simple, focused, how many hours did you work today kind of functionality and tomorrow maybe we add to it so think a little bit differently when you envision the way you're going to do these apps. One thing I want to do specifically is spend a second to talk about security, especially in the government marketplace. This is particularly important as it relates to the building app, when you're going to have government data on the app. I don't know if this slide has changed here but I'm going to assume it has changed for all of you. There we go. Any mobile application has double components and the architecture on iOS is understanding how it works when we consider how we secure data, I think that's a priority in any application that you build, so I wanted to mention a few characteristics you should keep in mind. In this little graphic you see just like any other information system, these information systems have three pieces, they have the data storage, the application logic represented by the icon here and memory where they store information while the application works with it. What's important about what secures iOS and to a lesser extent or to varying degrees, other plat tomorrows but we're talking about iOS today, this architecture allows us to write applications that persist data locally on the device that has application logic and manipulates a nonpersistent memory on the mobile device. These applications have a couple of characteristics. First, every app has to be digitally sound in order for it to run and that certificate that is used to find the app is generated and given

by Apple so if I'm a developer and I would like to write an app and I would like that app to run on an iPhone I would have to get a certificate from Apple by enrolling in one of the development programs and I have to sign that app. When Apple distributes the aptly their app store, they will tie it in again and that digital signature allows the app to run. It will not run on the phone without that. We're only talking about the standard authorized modes of the app, it won't run, so every app we can identify or can be identified who's the responsible party for that app and the app has not changed in any way since it was made so we know it did not have any other code inserted. Another important aspect of these app they are all sand boxed from each other. When your app is running on the phone, it cannot talk to the data sources and memory of other apps, nor can other apps talk to your data source and memory and this is important because it ensures that your app is safe if someone else writes a malicious app or your data is safe if someone else writes a malicious app. This is, for instance, why there is no concept or need for anything like virus protection on an iPhone because even if I could write a virus and even if I could get it on to the phone, it would not be able to manipulate anything other than itself and so it can't harm the data or application functionality of other apps and so there are some exceptions to this in the form of operating systems platform services like accessing the address book data and so on, but that's a controlled list of services that the developer is allowed to access but, in other words, your app would not allow it to do that, you actually can't. And then finally we have to think about data at wraps, how do I protect the data on the device? And that is decontradiction. By default on iOS devices devices all the data is fully encrypted all the time with a 256 encryption, however, that encryption -- literally everything is encrypted on the device but that can be turned off by unlocking the device and so for your application, if you would like to protect your data more strongly, you should leverage at the encryption capability, they call it date -- data protection, there are different levels and the highest form is data protection complete and this gives you options of encryptioning the data again, a much, much stronger form of encryption because it uses the user's pass code as part of the encryption key so if the device is compromised in some way, the data is secure. So the idea here is that security is about balancing functionality of the app with the privacy of the data and it's not hard to make a system so secure you can't use it at all, how do we balance it and make it useful? In deciding what kind of data you're going to develop to an app and how you're going to protect it, if we're talking about distributing apps on the iTunes store, in most takes, especially if it's federal data, we're not talking about highly secure data that we want to protect. In fact, it's the opposite. We want to get out there in the hands of people who can download apps in the store but if we're developing apps even through that methods that's accessing data in the back end, and many of you are asking me to do that, consider how you protect that data so it's not used in a way that's not intended. For inside and outside, security is a priority. Back to Steve martin, you've got your million dollars, now what are you took r to -- what are you going to do with it? There's several steps in the Apple work flow that you have to go through. The first of which is becoming part of a developer program and there are a couple of programs that you can enroll in, most notably the standard program or the enterprise program, and I'll talk about each of those in a second, and the device has to be provisioned. You have to get the devices to the people who need them, mostly for your developers because you're not going to be providing devices to citizens. You have to create an account, basically, in the iTunes app store infrastructure and identify that to your app and you have to submit the app to Apple for approval and the app store will distribute the app to anybody's phone and many of you probably have already downloaded apps to your iPhone from iTunes so you know how that works. Let's talk about the key develops as it relates to being a developer. There are two primary programs, one is the standard program and the other is enterprise program. The standard is for anyone who wants to build an app and twit -- distribute it on the I tune store, whether it's a game for fun or a federal agency developing an app for citizens, they all go through the iTunes store. Whether you charge for it or it's free, it doesn't matter. They all go through the same mechanism. The standard program is $99 a year to participate gives you everything you need to participate in that model, including tools for learning how to write apps, on line forums for interacting with other developers, a variety of other online resources for having what you need. Really they provide a vast amount of information that can give you-all the tools you feed

to write your app. There's also an enterprise version which is $299 a year and it is for organizations that would like to write apps that are distributed to their employees internally. These apps do not get distributed but the iTunes app store. They get distributes either directly to the device by an administrative task or more frequently in the federal government, through interm enterprise app stores, internal places where the phone can only list those apps you have written. If you were to write an inspection app for your inspectors or whatever, those you would distribute there that model. As part of the enterprise program, you have to identify yourself as the representative of that enterprise and so in federal government organizations, this can be nontrivial because are you the person who can be responsible for being the agency level party and make decisions on behalf of that organization? The Apple government team is available to help navigate that process when you become an enterprised developer, but in either case, you become the responsible party who is responsible for signing the app that you're going to distribute whether you distribute them in your enterprise or on the iTunes store. Another thing you'll get is access to the guidelines they have for distributing apps, which we'll talk a little bit more about later, but there are also guidelines for understanding not just what the app can do but what the roles and responsibilities are associated with being the administrator of a given program account. For instance, at Agilex we have a standard account where we can write applications and put them on the iTunes store but we do that with our own apps, not with those of our customers. We also have an enterprise account where we can write apps and distribute them to our employees but, as a developer of apps for other people, we also participate in their develop accounts so, for instance, if we work with the department of veteran's affairs, one of our customers, they join the enterprise program and they can have us be member developers on their program and so then we would write an application as a developer on their program and they distribute the app. They're the responsible party for ownership of that app and so the guidelines for how all this works are all available online. The main place to go is developer.apple..com. Many areas you can go, one of them is the member center. You can control who the members are on your program. I have an administrator who keeps track of what does what with it and we have developers who are member offense our program and they get access to our certificates and our development resources. There's a desk center, which is where they do all that and where they share information and where they get tutorials and documentation and all that and then there's iTunes connect, which is the interface you use ford to publish your application out to to the Eric -- iTunes store. Once you join one of these developer programs, then you have to build your app and the tool you use on the Mac is called X code. You get this for free when you download it from Apple's website. It's the same environment you get for writing Mac OS and iOS. The language that you use is objective C it's a variation on C or C plus plus. There's another component, which is the interface builder, which is a drag and drop tool for assembling a lot of the visual commons of app. You will see the buttons and lists and labels and so on using interface builder. You will practice your C experience and connect all those items up to each other to get an app that does something and I think you'll find that if you have the programming or java or C plus plus programming experience, you can very quickly get to a working app. What's also important is that Xcode can become the interface to which you publish up to the app store as well. In a setting configuration, you have the account that you have for the development environment and Apple.com and when you do a build of your application, it can build into that environment and do all the appropriate applications and so on but it becomes where you develop your iOS app. I mentioned this already but there are a vast number of resources available oncer logged in to the environment. As an aside, Apple is pretty rigorous in making sure that you follow their rules and unof the -- one of the rules you have to follow is that you don't share developer information publicly unless they're developers, right? So technically as a developer, a member of the development program, we're not supposed to publish information that's on them, so these are just screen shots at the high level of stuff you see when you're not logged in, but if you go to developer.apple.com, you can see a lot of information and once you become a member, there's a ton -- trust me, all that you need is there in order to see detailed technical information as well as support groups and documentation and so forth. A couple of things about actually building your app. First, the guidelines that Apple puts forth on what an app can do are not really controlled by the device. It's not

so much what kinds of apps you're allowed to write for an iPhone, it's really what kind of apps they will allow you to sell in their store so these guidelines, think of them more about controlling what apps are appropriate on the store more so than what you could do on the phone, and so the reason I make that distinction is if you think of them as a retailer, as a store that will distribute your app for you, they control what they think is appropriate to be in that environment; however, if you need an app that does something and you only want to run it on your phone, you can write an app that does whatever you want, as long as you don't distribute it through their store. Or on the enterprise part of it, you can write an app to do whatever you want. There are security restrictions on the phone but as far as appropriateness and so on, you can do whatever you want and distribute it directly inside the enterprise but if you want something out on their store, you have to follow guidelines. Once you become a member of the program, you can download this list of guidelines and it's pretty straightforward stuff. A lot of it is obvious. They don't sell apps that are inappropriate for children or have porn in them or they are increasingly talk about apps that are useful, apps that don't do anything or are a waste of time or that there are 400 apps in the store that do the exact same thing, there's a little bit of that. There are some technical things you can't have an app that calls unpublished ATIs. You can't have an app that, for instance, using the camera unless it is obvious to the user that the camera is currently being used, right? So you can't write an app that spies on the user somehow and so on. So there's a long list of guidelines of things that technically your app can or can't do and I would put them in buckets, like I said, the first is about appropriateness and second there are technical restrictions, most notingly, calling API settings that are not published, they say you're not allowed to call those. How they go through the process of checking that is twofold. They have people that review each app, but they don't look through your source code, they review all the materials you submit about the app. When you submit the app you put in what it does and I -- icons of the app and the link to your developer website and so on and so forth. People review that and they get what your app does. There's a technical vary occasion, they confirm that the app does not call on APIs, which they can do in an automated fashion. If they're cool wit, they approve with it and that approval process is not a long one. Typically it just takes a couple of days, as much as a week, unless you're high profile and doing something that is really newsworthy and they have to think long and hard whether or not they're going to allow it like when Google first came out with their gag t voice app, Apple has hesitation about whether or not they were going to allow that but most of us are not in that situation. We had an app that we submitted and they came back and said this is a beta version in your description and we didn't need to so just a little give and take. They're just checking all the Is and Ts to be dotted and crossed and they distribute it through the app store. There's another thing I mention, it says here the car wash. Apple also, for high profile apps, will offer putting it through a review process by a team they have that does user interface review, don't say oh, gosh, that's ugly, you should put this image in the background, it's mostly about consistency and usability and they say stuff like oh, when you tap here, the whole page flips over but that flip over should be for showing settings of what was behind t rather than for navigation, they just get consistency for the way things look. They don't do this for everybody. They don't just do it because you ask. They do it for those apps that are going to be high profile and that merit the time that it takes to do that. We have gone through this process for some of our government customers and I'm sure if you are writing an app that is representing an agency and you think it could benefit from this, it's something you should ask Apple about making available, but like I said, it's sort of a nonadvertised extra bit of help they give just to improve the quality of the app that's out there. So then once you submit -- to submit you app, you do it through the iTunes connect interface. This is the process by which you take all the screen shots, you take the description, you fill in what countries the app can be downloaded from, what can I charge for the app and so on and you upload it into the portal they have in developer.com and then you wait for them to come back and say you have a misspelling in your description or your app calls an API that's prohibited or this app is useless or we love it or whatever, usually within a week they will say your app is approved and then you use iTunes connect for a different purpose, which is really now about monitoring your environment. This screen is just a sample graphic from the web that is the kind of monitoring you

might get, how many times my app has been down loaded, in what countries, how many money have I made and so on and so fort -- forth. I think it's as important to instrument your app so that there are different components of your app or if you're trying to get usage data, that's going to be as much about instrumenting your app as it is about expecting that iTunes will give you that, this is more about the retail side of distributing your app by it's an important way to keep track of how your app does. With all that in mind, I thought I was give you a specific example of an app we did like this for the government. We don't really, in most of the application development that we do is focused on helping our customers build enterprise apps, exposing their internal systems but we do have a few customers for whom we have helped build apps that are distributed through the model we have been discussing, through the iTunes app store, most notably is Amtrak. We have built several apps for Amtrak. We built, for instance, the enterprise app are conductors are taking tickets on the same that accesses their reservation system. As I mentioned, the way they get disconnected and secure and all that stuff but that same app has a component that riders or passengers of the train can buy their tickets and check the status of a train and this is an app that they distribute on the iTunes store. If you were to go to iTunes right now and search for Amtrak as I did, you would see this information, the description of it, what's new in the version, screen shots of the app, who built it, how big it is and so on. Actually, we don't say who built it. It says the national railroad passenger corporation. That's the component of Amtrak, the technical organization Amtrak that's responsible for this app even though we were the developers of it, we were doing it on their development program and like I said, that's the typical way that you can get others to help you. So this app, then, gets downloaded by folks like you and me and we buy tickets with it. We do use iTunes connect to monitor that, how many times was it downloaded, that kind of stuff, but the real use comes in the app itself when we now know that somebody bought a ticket from the phone instead of from the web or instead of from the ticket counter to get, what I would call, more business oriented instrumentation to understand the value of having a mobile app like this one. This app has been astonishingly successful. I don't know if you say a i New York Times article even this week but it was talking about Amtrak doing this on the conductor side and on the passenger side, both accessing the same back end federal information system in the form of a reservation system and that popularity has shown them a great deal of how they can advance their business models by taking something that they have been doing all along and creating a mobile version. This is an app, also, where the user experience has been really important. We wanted it to look and feel like a traditional iPhone app, that people now how to navigate it without any training but we wanted it to be pretty and interesting. You can't quite see here because this is a screen shot but I was able to scroll it to the right, once you get into looking at the configuration of the sleeper car, the train your on, there's cool stuff and there's a program where kids can keep track of the train stations and so on. It's focused on what's enabled by mobility, so I wanted to felt you a little bit of that tale, the development of this took a while. We had a distributive team of folks from various companies helping from user interface experts to back end systems experts, some mobile development experts and we ran that team, then, to not only get the back end ready so all of the web services required to interact with that back end reservation system, having them exposed to the internet in such a way that they could be accessed by anyone as well as in a secure fashion complementing what they have been doing already on their website and so on. The actual building of the app is not the heat index lift in this kind of thing, it's that whole system being created in a secure way but building the app and testing it and getting it through the accreditation process that has to happen inside a government act or omission -- organization of it but the submission of the app through the store was not difficult at all. I thought you might find it a useful example of one of the commercially available apps through a retail store like iTunes. I also like that it has a sister app that is not commercially available that the conductors use and they interact with the same back end data so for us it's been a great test, an ecosystem of the kinds of issues we encounter in building an app inside government. With that in mind, I will draw my prepared comments to a close. My hope was to tell you a bit about the process of building an app. I didn't go into much detail about how to build an app or what your app needs to do or things like that. We do run webinars on those kinds of topics. We're having one next week, you can check our website if you're interesting but if you want to go deeper in the

technical issues, we can do that. Mostly we wanted to focus today on a general overview of the Apple process for building and distributing those apps but with that in mind, we'll take questions if there are any.
>> Thank you so much, Tim. That was great. There are a few questions if people want to keep bringing in questions, we have plenty of time. I'll start with what you guys are asking. This comes from Mark. What are the best practices for posting the app source code and federally funded work is not copyrighted, what are the processes on that?
>> Our experience has been that even though source code is not copyrighted such that you would keep it hidden from folks, we don't freely distribute it. Sorry, I'm hesitating because I think there's two answers. There's sometimes when it serves the public good and think of this as the way the open source community works. If you build something cool and you would like to distribute that to help enhance the community of government folks who leverage that code, I'm all for it, but just because you have a mobile app that the government wrote and owns does not mean they have to publish this source code. I'm not even sure the regulation can demand that you get access to source code. In my mind, that would not be something that we would typically do, mostly because it would make it easy for people to try and find as a rule -- vulnerabilities in that code. You don't give the source code to Apple. They don't have it and publish it, they don't review it. My guess is part of the agreement suggests that they can request it and if for some reason they want to review it they can, but typically they don't. They just run automated tools against the executable code to make sure it doesn't call any ATIs are are not published but the source code itself, ordinary other than the community good that comes from open source sharing, we could not advise our customers to share that code. There are several moves afoot inside government to create mobile centers of excellence where make in a department working across agencies to allow code templates to be shared and so on. We do have, I think with maybe two of our customers where we have been suggest up those environments where there's build infrastructure and it's build in a certain way, integration is automated and there's also code repositories so people can share code been by that's different than sharing code publicly or commercially. I'm not sure if that answered your question.
>> I think that's on the right track and there are other initiatives that are coming out in the federal government and we'll probably offer more guidance about this. A lot of questions coming in. If you could keep asking your questions, folks on the webinar, there are folks here that are sort of curious about -- they're asking about the metrics once the app is developed. Publishes complaint that Apple does not provide any data about the behaviors within the I tune store, I think folks are looking for are what are the analystics that people are looking for?
>> Most of the analytics are about the distribution of the app, how many people have downloaded it, no -- not who specifically has downloaded it but from what country has it been downloaded, who paid for it, the tools are targeted more for somebody who is selling the app on this store and wants to foe how they to -- know how they're doing. If you would like better metrics about how your app is used, our experience has been that we instrument that into the app itself. When I mentioned that Amtrak can very accurately assess what percentage of their customers are buying their tickets through the iPhone app as opposed to through a web browser on the iPhone or on a desktop, all of that data, which is quite valuable to them, they can say gosh, the people who bought their ticket on the app made fewer calls to support to get the ticket changed. All that kind of data is inside the app. It's not the kind of data that the iTunes store would distribute. Now the question also maybe is referring to, I thought you said publishers as well, which, Apple's got a content distribution arm as well in the form of music and books that they sell so the publishers are probably quite interested in who's downloading their books and so forth but from an app perspective, it's predominantly about the business side of it.

>> Thanks, Tim. People were a little bit interested about developing Apple apps internally like how you talked about the Amtrak app for conductors. Can you spend a minute or two about how that process might be different from the customer focus process?
>> Sure, there are a couple of things. The way you build an app for the phone are exactly the same. You build the same tools, code the same and all that. The primary differences are first you have to be an enterprise, a registered enterprise. When

you enroll in the enterprise program, you provide the number of your company and you are to value data that you are a real company and so on. Most of the peoples listening are not companies, they're agencies, so the process of doing that is a little bit more involved because the person who signs up has to be able to speak on behalf of the agency as it comes to signing documents and so on but, again, I mention that had the Apple federal team knows how to take you through that process if you need to. Once you're on enterprise developer, a couple of things. Oh, one thing I didn't mention about those prom r programs and it's relevant now, when you write your app you want to test it and you want to test it on a phone. There's a similarity that similar -- simulator on X code that allows you to test your app on a simulated iPhone and iPad on your characteristics but you also want to test it on a real phone. You are allowed to find up too 100 devices for each program that you can install directly on the device from your Mac. Basically I write an iPhone app, I plug my phone it, it December do you want this phone to be one of your 100 phones, not just any phone, this particular phone or iPod Touch or iPad gets registered as a device that can run applications that are signed with a certificate that you get as part of your developer program. It's very specific and you have have up to 100 of those. These are used for developers and for testing so you're restricted as a standard developer, you can't install this on more than 100 devices without going through the iTunes store. With enterprise you can install on any device and you write the app, you build the app, you sign it with your enterprise certificate inside X code and then you take that app and you can install it on any device, the idea being that you could install it only on the device of the employees but you don't have to know that particular device. There's an identifier called the UDID which stands for, I don't know, universal DID or something. I don't know. It's the identifier for the device and they are unique with every device W. with the standard program you have to have that but the enterprise program you don't. Now imagine I'm an enterprise and I have 1,000 people I'm going to give this app to, I can just send it to them and they can install it on their device. So what's to keep you from just writing an app from your enterprise program and put it on your website and letting anybody install it on their device. If -- that would revoke your certificate and that app wouldn't work anymore so that's a little bit how the control mechanism works. That maybe distinction dis -- main distinction is I can write an app and distribute it too my -- to my employees so it might do something -- well, most likely it's going to access some internal system. These iOS devices support the common DPN so maybe my workers are in the field and they sign through VPN. We have an application that we build for the VA where doctors with their iPads and iPhones are accessing the electronic medical regards of veterans while at the bedside so there's an information system at the back end, the networks from which they're allowed to do it is very carefully controlled, the devices from which they're allowed tide are allow -- to do it are controlled. There would never be such an app public by but even if they did, it wouldn't work publicly because you have to be within these accounts on these networks, so it gives you much more control over what you do and then it's really just a matter of distribution of those apps, typically, like I said, we build internal app stores where people can download the apps to their phone, or you use -- when you deploy apps on any mobile device inside an enterprise, you're going to use management software like mobile device management software and those can also distribute apps to your phones or give you a store front where you can download them. I don't know if that answers your question, but hopefully it does.

>> I think that's great. We have time for a couple more questions. This is one -- I'm going to ask you two relatively quick ones and I have the one I want to end it on. How can you ensure and implement security through authentication for applications built for government?

>> Clearly inside the app you want to have authentication. There's authenticating for the device and your mobile device management can force strong forms of authentication just to unlock the devicement when you launch the app, it can authenticate as well. With any kind of authentication methods you want to write, user name and password or token based authentication. It can read a PIN card or we have a customer for whom we built a biometric method so your app can be built with whatever tools you want to put in your app. There are not adds many tools available to unlock the phone. You can make sure the password is 17 characters and has so many letters in it but you can do whatever autothen occasion you want to once you build

your app.

>>> Thanks. So there are a couple of questions we can go a little bit longer, I have been granted a little bit of time so if you have questions, put them in. There are questions around, Tim, about the technical expertise required to build an app. Do each developer who accesses the resources, would they feed their own standard program account? What's the nature of the technical expertise needed and how many developers can work at a time?
>> Let me answer the second one first. Not every developer needs their own standard program. An organization would get a standard program and they can invite members, developers to participate and do development with the certificate on that. There's also an administrator who controls whether or not those apps can be distributed. For instance, Agilex has a standard program. We have dozens and dozens of developers who are members of that program but I am the agent for Agilex as we work with Apple and only I could take one of the apps those people develop and pun lush it on iTunes for Agilex's behalf. Those developmenters can test them all they want but they cannot distribute them but for all that, we pay $99. Number two, what skills are required? Well the skills required to get an app to work and then there are skills required to get an app to be good and first is straightforward structured programming experience in -- the fee is good, objective fee, it's a variation on some of the syntax a little bit but any java or C programmer with a book and a long weekend could be a programmer but to make an app good, there are two other kinds of expertise, one is user experience, and this is not just graphics experience, but how an app works and we have some brilliant developers but they're not -- and often good developers are votive problem -- problem solvers but we have creative people who are not necessarily understanding of how to use scarce resources. Most of these mobile devices, applications, if they're interesting are going to access some type of back end system and for that you need an additional set of knowledge, maybe java skills to write the web service that is you will access from the mobile device, people who are familiar with accessing databases and doing information security, not really networking expertise but more how I opt muse information across scare network resources, disconnected applications required, different levels of skill and so on so to build a real compelling UI, compelling and high performance functionality and accessing a back end enterprise system is leverage you a lot of those skills but the barrier to entry, as evidenced by the hundreds of thousands of apps out there that teenagers have written in their basement, you can get a very compelling app very quickly, but a government app that accesses a system in a secure way requires those higher orders of skills.

>>> Thanks, Tim. What would you say the typical time frame to develop and publish an app would be?
>> We get this question a lot and I would say maybe this re relates to the last question. Just because an app costs 99 cents doesn't mean I can build it in a day. Most of the app that is we build, we have a working prototype in a matter of days but getting it -- and that's like screen navigation and how it might flow and so on but getting it from there to ready to be distributed to thousands of people can take weeks or months, but a lot of that is not in the app, it's in the back end, getting those web services written, making sure they're secure, on the musing data -- optimizing data access because remember, it may be designed with a small number of web users in mind internally and now all of a sudden you're going to expose it to shows of people who have very small transaction that is they issue many times rather than the big reporting it once or whatever, the characteristics of how those apps are developmented matters. One challenge as we help educate government organizations on what it takes is they think it should take a long weekend and then it's all in the app but in fact, it's not much in the app and it can take as long as it would take to build a web information system if it's complex. Did that skirt the question?

>> No, I think that's an accessing rat porytrial of an answer. Another question we have to be delicate with. I think it's related to the answer you just gave. What are the ballpark figures for development costs? Is there anything like that? Have you seen an average? I know when I talk to agencies there have been various numbers thrown around. Is there anything that feels right to you or how would you suggest agents approach financing it?

>> Well, I would say the best -- I'm not afraid to answer that but there's no one answer. That's why I hesitate a little bit. If you have simply a stand alone app that does one thing and it's all the self-contained, you create an app that contains the locations of, I don't know, national parks, and there are only so many and we know where they are and you put it all in the app, you're not going to show it on a map, you're just going to list them and show where they are and whatever, just a simple self-contained thing. That won't take that long to build, days to get it built and weeks to get it approved. Multiply that by what you pay your developers and you can get a simple stand alone app written for let's say tens of thousands of dollars. Sorry, my phone is ringing. Bear with me for one second, I'll just talk over it. Maybe I'll just hang up on them. Mercifully I could tell who it was, hopefully he's not listening. But as I mentioned earlier, other than those apps, which I frankly think are the exception, not the rule, I would use the same guidelines that you use to scale and size the cost associated with building other enterprise information systems. What does it take to prepare the database in the back end? What does it take to buy software and hardware for native apps data environment? What does it take to build web services and access that data? What does it take to design and build X number of screened apps. One additional cost is maybe graphic design help to make it truly compelling and add up the cost that way and then it quickly gets into six figure engagements and for big, large programs, organizations will spend seven figures because that's what it takes to get that kind of information securely to that many users and in some cases, that's cheaper than what it took to build the client server or web version of the same kind of system. If you're talking about an app, it's not that expensive. A system can be costly to do properly.
>> Great. Okay, I have one more question for you and this is going to take us a little bit away from talking about iOS here. What are the main advantages of building apps in a native environment versus one in a web based environment?
>> Okay, I would say there are, let's call it four different buckets of how you might do it. One is native, whether it's iOS or Android or whatever. One is HTML5, which says it's web based although there's functionality running on the phone but it's predominantly a fancy web app. There are MEAP, a mobile enterprise application platform. There's one called MMEAP and then there's VDI, virtual desktop and you're providing virtual access to it and some enterprises are using this as a way to quickly get access to their mobile uses. This is a much less prevalent way of doing this. You can download apps now off your iPad that allow you to access Microsoft office online. I can basically, in a virtual chillized environment, use a cloud version of Microsoft so office. Let's not talk anymore about that one because if you have an existing app or you would like to virtualize access to your database, that's I -- well, I think the end result would be quick access to the functionality you need but not a rich mobile experience because those U Is were designed far mouse, -- for a mouse, not a finger. The MEAPs, there's a wide variety of them and they are all over the price in their maturity. This is a movie we have been to before. We tried to do this in the client server world, in the web world where we build a single GUI. My feeling on MEAPs is on general, you end up building a least common denominator so while you can get an app that works, that runs on many platforms, it doesn't leverage any of them to their full capacity but for an IT.com, that might be swell, especially as organizations are thinking about bringing their own model but your final user experience would not be as great. The same is true but to a lesser extent for HTML R5. The development skills are different for java and java script and the user experience you can get is compelling; how far, it will not necessarily look or feel native because if you make an HTML5 that looks native on an iPhone, when you run it on Android, it will look like an iPhone app. HTML5 does not require that you are connected. It can run in disconnected mode, we can persist data in HTML application on the local device, however, I still think you will end up coding a little bit and user experience can be very good but it's not quite like the native feel and then finally native application development is the richest in user experience. It's most time consuming but to be frank, 90 percent of the functionality should be in the networking anyway. The app should be just a user experience and if you do that well, and you get that user experience layer very thin, then it's not that hard to write an iPhone and Android version of that user experience. We had one customer who asked the exact same question so we're taking an app and building it five different ways, exact function that willty with all five

different models to show what it will look like in each model and we're learning once we build it for iOS rebuilding it for Android, it's not that difficult. The heavy lift is the network. I think the MEAP will get better and HTML5 will always continue to progress. My personal experience is that if you can do it native, everyone if it means writing it a couple of times, your users will be much happier.

>>> Well, Tim, thank you so much for your interesting insights today on Apple and on mobile development in general. A couple folks did ask, they wanterred if they could get access to the webinar that Agilex has every week.
>> We don't do them necessarily every week. We do have one next week. If you go to the Agilex website there's a webinar on there that we scheduled for the 16th, or if you want I can give you an e-mail address that you can check in with Kristin. It's kristen.kessler@agilex.com and she would be happy to take requests for information. The one I'm talk about is on the 16th and we're going to talk about, this more detail, the issues associated with app development.
>> Great. Thanks so much again. Folks, thank you for attending today. I want to remind you that if you have questions about mobile development in the federal government check out the Mobile Gov wiki. Also, we have webinars for our windows phone and our blackberry platform seminars that we did in the series, you can find those on the registration page to this webinar. Also, this webinar will be available for you in the next few days where you can come back and relisten and hear what Tim has said. We will be doing an Android series here in a few weeks. Say tuned for the date and time for that. Otherwise, have a great day. There are other digital gov university webinars, the next one is May 24th about minimizing security risks while using social media. You can find all that on hOwto.gov. It's been great, we'll talk to you again soon. Good-bye.