071417 Air Force Association Mitchell Institute for Aerospace Studies Space Power to the Warfighter Seminar with General Ellen Pawlikowski, Commander, Air Force Materiel Command, on "Keeping a Decisive Edge: Agile Software Development in the United States Air Force." (For additional information on NDIA/AFA/ROA seminars contact Peter Huessy at phuessy@afa.org).

MR. PETER HUESSY: Good morning, everybody. On behalf of the Mitchell Institute for Aerospace Studies and the Air Force Association, my name is Peter Huessy and I want to thank you for being here at our 29th seminar this year on space as well as missile defense and nuclear deterrence issues.

Just a few things. Next week we hear from Ambassador Ron Lehman, formerly of Lawrence Livermore National Labs, on the 20th. On the 25th Senator Joe Donnelly will be coming to speak to us. Our honored guest today, Mike Rogers, is going to be speaking to us on the 28th of July. Then we hear from General Selva on the 3rd of August. Uzi Rubin ends the summer with an address on the 11th of August on the Iranian, North Korean, Chinese connections on ballistic missiles.

This is the third year we've been doing our Space Power to the Warfighter series. Our guest today is General Pawlikowski and was a speaker, I believe two years ago.

I want to thank our embassy guests that are here, as well as our military service men and women who are here. I want to thank our sponsors and our supporters. And I want to thank everyone who makes these series possible.

Today Congressman Mike Rogers is going to introduce our speaker, General Pawlikowski. As you know, Mike Rogers has been a long-time speaker at this series. He was elected from Alabama's 3rd District in 2002. He serves on Agriculture, Homeland Security and on the House Armed Services Committee where, as Chairman of the Strategic Forces Subcommittee, he has put both missile defense and nuclear weapons, deterrence and modernization issues in the right direction. Those are my two god children, if you will. Congressman, you have done yeoman's work in this area, as well as trying to get our space assets modernized and funded correctly.

With that, would you give a very warm welcome to our wonderful friend from Alabama, the Chairman of the Strategic Forces Subcommittee of the House Armed Services Committee, Congressman Mike Rogers?

(Applause).

REP. MICHAEL ROGERS: Thank you, it is my honor to be here this morning to introduce the speaker. Ellen Pawlikowski at Air Force Materiel Command is just an American treasure. One of the benefits of being in office as long as I have -- I've been in office 31 years -- is I get to meet a lot of very successful people, people who are leaders in business, people who are leaders in government, and obviously people who are leaders

in the military. The problem with that is after a while it's pretty hard to impress me.

(Laughter).

But I can say from the day I met Ellen Pawlikowski she has been nothing but impressive. She has been a winner in every job that she's had, does an outstanding job for America. And if she wasn't doing such a good job at Air Force Materiel Command, I may get here to run Space Corps, but I won't do that.

(Laughter).

She is too valuable where she is. I know she is going to do a great job enlightening us on what we need to be doing going forward in the Air Force. So join me in welcoming a great American, Ellen Pawlikowski.

(Clapping).

GEN. ELLEN PAWLIKOWSKI: First of all I want to thank the Mitchell Institute for inviting me to talk. It's always great to be back. When I got the call and they said we want you to talk at Space Power for Warfighters, I was initially like, wait a minute, I don't do space. But then I looked at the folks that were going to attend and I said I can't say that anymore.

(Laughter).

You all probably remember that when I first went out to work MILSATCOM I said I didn't know anything about space. Then when they sent me to the NRO I said I really still don't know much more about space. And then they sent me back out to SMC, so I guess I can't say I don't know anything about space anymore.

(Laughter).

It's hard for me to believe, but it has actually been three years since I left SMC. I feel kind of -- not to toot my own horn -- but I actually felt pretty good leaving SMC that I had left things in better shape than when I got there and was pretty proud of our successes. When I looked at the fact that we now had the SBIRS constellation, not just one satellite, not just no satellites like it was when I left; and we had Advanced EHF flying, and close to an initial operating capability, and wideband global SATCOM was now hugely successful.

But I have to admit there was a little part of me that was bothered, in that there was a couple of key programs that despite all of my efforts to try to make successful, I left without feeling comfortable about where they were. Those two programs were JSpOC Mission Systems and OCX. Sorry, Lynn.

The thing about both of those, the commonality about both of those, was they

were key software programs. The other thing about both of those was that both of them had been touted as applications of agile software development. That's how we started them, with great enthusiasm and excitement. Three or four years later we were looking at huge over-runs or delays or just ineffectiveness in some way.

So I kind of still kept that in the back of my mind, but I thought well, okay, on to a new job. I won't have to be worrying so much about software development in this job because it's an organize, train and equip job. Then we got a new chief, and I have to tell you a really exciting chief because he gets it when it comes to war fighting. He laid out a vision and he said the successful commander in 2030 is the commander that can command and control multi-domains, providing effects from different domains at the speed at which our adversary is overwhelmed and not knowing where the next attack is coming from, at the same time we deny him the ability to do that.

I thought about that and I said, boy, we're going to need a lot of SATCOM, when you think about that. We're going to need to be able to quickly change and adapt to be able to do that. And then he came to me, as the organize, train and equip or what you might say is the senior uniformed person in acquisition.

He said, Ellen, I need your help here because we have to rethink about how we do acquisition. We need to think about not stovepiped pieces of hardware, but we need to think about a network of apps and apertures. So I'm thinking to myself okay, we're going to need that comm again.

But oh my goodness, remember that agile software development stuff that I kind of was frustrated that we hadn't been successful at? If I can't do software with agility and the ability to be able to think about this as a network, think about that. The heart of it is going to be -- of course you know the physics, the engineering.

He said, I need RF. The network has got to be able to carry signals from one place to another. But the heart of that is that brain trust called the software.

So it became one of my personal things to go back and look at this agile software development again and ask myself, what happened? Why were we not able to implement that? And by the way, I learned as I got to AFMC, that the space business was not alone at being able to apply agile software development.

This was back on my forefront of my activities. And then, of course, along with this charter that General Goldfein gave to me in terms of looking at how do we change the way we think about our acquisition, he also charged something called the multi-domain command and control enterprise capability. And if you talk about that adversary, you're talking about multi-domain command and control, and I've got to talk about integrating a bunch of stuff together that I haven't really done very well in the past.

And guess who he put in charge of working that multi-domain command and control? He put a space officer in charge of it, because in the space business we

understand that better, I think, than many of the other domains do because it is so critical to us because of the distance between our command and control capability and what we're trying to command and control. Think about it, right?

We worry in the air business about using beyond line of sight comm. In the space business, we don't even think about line of sight comm, right? It's all distances. So it all starts to come together that the ability of us as a Department of Defense to use, to be agile in our software development, is critical.

Then add to that the growth in the cost associated with software when it comes to our weapons systems. The first place this hit me, by the way, was when I was working MILSATCOM and we started to look at, with Al Ballinger (ph), the amount of software code on GPS-3, the code on-orbit, versus something like the original DSCS satellites which were bent pipes, or even the first GPS. And you realize that software is at the heart of most of what we do.

So I'm kind of a back to basics kind of person. So I said alright, if I'm going to understand this agile software development I need to understand what it really means, because it isn't a buzz word. Everybody loves to throw it out there, right? But what is it really about?

So I actually did a little bit of homework and it turns out that it started back in 2001 when a bunch of software development minds got together at a ski resort in Utah. They said were trying to steer the way we do software development away from the traditional way, which was documentation driven and heavy software development processes. They said, okay, we're going to establish 12 guiding principles.

I thought, this is good. I can compare their guiding principles to what my objectives are when we talk about providing war fighter capability. The first one is our highest priority is to satisfy the customer through early and continuous delivery of valuable software. I liked that one, right? Who could argue about that one for anything we do?

The second one is, welcoming change in requirements, even late in development. Agile processes harness change to customer's competitive advantage. What does that mean for a traditional acquisition in our mindset? Stable requirements, that's what we need, but this one's a little different.

I will tell you that in my experience stable requirements are non-existent, as far as we try. So this is saying -- you know, I have a black belt in Taekwondo and we always talk about when something comes at us in force, how do you use that force to your advantage as opposed to trying to oppose it? Well, this says take the fact that requirements change and use it to your advantage. So I liked that one.

The third one was deliver working software frequently from a couple of weeks to a couple of months, with preference for shorter time scales. Well, okay.

Business people and developers must work together daily through the project. I like that one. I'm not so sure we do that, do we?

We've got multiple layers between the guys actually going to operate, when they say by business people, and the guys writing the code, don't we? Even within the government we've got operators over here and acquirers over there. So that one may take a little work.

Build projects around motivated individuals and give them the environment and support they need and trust them to get the job done. Trust, where have we heard that term before in terms of acquisition?

The most efficient and effective method of conveying information to and within a development team is face-to-face conversation. I like that.

Working software -- this one is the one I really love -- working software is the primary measure of progress. Yeah, I love it. I don't care how much paper you've got, give me a piece of software that works.

Agile processes promotes sustainable development. Sponsors, developers and users should be able to maintain a constant pace indefinitely. A constant pace, not build up to a milestone and then build up to another milestone. Just keep on going, keep running. As a chemical engineer the best thing you do is to have a steady state process, one that doesn't start up and stop. And shutdowns are the highest risk, right, when you're doing that.

Continuous attention to technical excellence; quality, right? Mission assurance for the space guys, right?

And good design enhances agility. Simplicity, keep it simple. The art of maximizing the amount of work not done is essential. I love it. The best architectures, requirements and designs emerge from self-organizing teams. Remember those IPTs (ph) we used to talk about and pulling them together to have the right people?

And then finally, at regular intervals the team reflects on how to become more effective and then tunes and adjusts behavior. So this is the kind of process for agile software development. You plan it, you build it, you launch it and you get feedback. And you do this constantly. You keep that thing running and you think about that wheel running.

Go to the next chart. Then I started looking at who has done this really well and been very successful. Well Google, for example, they obviously live on software and they do software releases multiple times per week -- multiple times per week. Think about us with the product cycles that we have.

Facebook users upload more than 2.4 billion content items daily, and their software engineers are delivering changes to their production environment. All of us that are Facebook users have no idea they're doing this, right? They are doing it 50 times per day. They're making software changes 50 times a day.

You might say okay, that's only information, social media, who cares about the risk? John Deere recently said that they were going to adopt this. In 2012 they actually hired 800 software developers that use an agile process. They use this as a way to be more responsive to their customers. They're in tractors. They recognize the value of software.

Amazon went on record in 2011 that they were going to release software updates every 11.6 seconds. In 2014 they deployed 50 million software changes. That's one software release every second over the course of the year. They do it, and we never see -- how often in the last year have you known that Amazon totally crashed?

At TomTom Navigation, they took the fail fast approach and they're producing software every two weeks, one to two million lines of code per navigation product. They have 750 to 1,000 engineers across 11 development sites.

So you look at this and you say, there's obviously success here. These are all companies that have been very successful. So why can't I get that success?

We've got some examples within the government where we have done agile software development. I think under the previous administration we really embraced this DIUX and Defense Digital Services. Frankly, a lot of what DDS is trying to do is to introduce software development.

But now the next step is okay, I want some of this. I've seen it successful. I like the basic tenet, so why can't I do it?

So I go back, and Lynn is sitting here and she will resonate with this one. When I looked at all the things that we were doing on OCX, on JSpOC mission systems, on AOC 10.2, which is an air one, I have come up with this set of what I call the barriers that are frankly 90 percent cultural, five percent things that DOD did to themselves, and five percent of what Congress has done to us.

So let's talk about this. The first one is the one that I think Lynn will relate with me. We are very much enamored with our system engineering processes in the Department of Defense. Systems engineering V, by golly it works, we're successful, we have processes that drive us to go all the way down starting with requirements, and going all the way back up and rigorous testing.

Well about the second week I was on the job at SMC, I learned that OCX had failed their preliminary design review. I'm like, oh my gosh, how could you fail a preliminary design review? I mean, that's one of the initial steps, except that preliminary

design reviews don't make any sense when you're doing agile software development.

Remember, agile software development is all about getting capability out there. The systems engineering V drives you to this detailed requirements flow down and then you write the software development process. What is the PDR for? Is it for that first scrum, that first rapid fielding of software?

So the first thing that we need to do, or we could then go to what we call the water fall model for software development, which is another one. It's this very structured way. And in these cases you don't start writing code until you've already wrote sometimes thousands of pieces of paper. You do this detailed requirements flow down.

Remember tenet number two that requirements are going to change? By forcing our software contractors, and by implementing this into these models for systems engineering, we have essentially negated two to three of the basic tenets of agile software development and make it very difficult for our programs to succeed.

So barrier number one that I think we need to do is we have to change the way we think about systems engineering. We have to change the way we think about requirements definition as we're going to really adopt agile software development. Maybe the answer isn't this detailed requirements flow down.

If you think about it, you want to deliver working software fast, maybe we slice it this way instead of digging deeply into it that way. That's really what agile software development folks do. They continue to field capability and they do it by -- maybe all the requirements aren't met at the first one, but you have something that functions, that you can put in the hands of the operator and they can use it.

And oh by the way, once you put it in the hands of the operator, think of some of those requirements you had in the beginning, maybe they don't make any sense anymore because the operator sees how they can actually use this and they change it. Ok, so that's the first one.

The next thing is, you remember that thing about putting the coder and the user together? We don't do that either. We have an industry guy that sits over in his facility in Colorado, and then we have the operator that's trying to do this in Colorado Springs, and we don't let that operator see that software until it goes through development tests. Then it goes through an independent operational test, and then we finally let the guy who is actually going to use the code get his hands on it. Well that violates the other tenet of let's get everybody together, that face-to-face communication, and turn that cycle fast.

Here is where I remember when I was working JSpOC mission systems; first of all you have to have flexibility in requirements, right? JSpOC mission systems recognized that when it was first started. In fact, the original premise was that Lieutenant General Susan Helms, as the primary operator; and Lieutenant General Ellen Pawlikowski, as the primary acquirer, would share this requirements group. We would

make the decision on which requirements were going to go into essentially the next scrum, if you will, the next release. And we were going to release software -- I know this is slow for agile software -- basically at six month intervals.

So somebody decided no, no, no, no, this is a high priority program, a really important program, it's going to be called an ACAT-1 and therefore requirements have to be run through the JROC. Oh my God, what happened to that? I just shook my head because now we have a three-star who's in charge of space operations and we cannot trust that three-star in charge of space operations to determine what the requirement should be. And we can't trust the three-star in charge of executing it to know whether we can execute those or not, because we now had to go through OIPPs and DABS for the decisions.

So we violated the basic premise in that one. Then, when we sat down to say okay we've got to test this stuff, our six month scrums -- remember we're doing this to deliver in six months -- AFOTEC said it was going to take them 18 months to do the OT and produce the report and provide their recommendation. So we have to empower at the right level, and that has to be at the level of the person that's going to use the software.

And we have to stop thinking about independent OT. What does that mean? It just doesn't make sense when you think about being able to agilely be responsive with software.

Remember those timelines. We need to be able to provide multiple effects from multiple domains at the speed that totally overwhelms our adversary. That means that General Saltzman's multi-domain command and control system has to be flexible enough to be able to adapt the tools it has, which is the software. It has to have somebody sitting side-by-side with them to be able to make adjustments.

In the space business, some of you will remember way back when it was engineers that were sitting at those command and control terminals, and writing code, back when. So we have to get those barriers out of there. We've got to do it.

The other issue is we hold onto stuff too long, and then we end up having to try to figure out how to connect a dinosaur to a jet airplane. We slow everything else down instead of just saying -- and an example of that becomes AOC 10.2 where we have all these multiple stovepiped software pieces that we want to keep. When you start to look at what happened to that program, we ended up with your system being totally bogged down because every time you made a change there was 50 different stovepipe priority software pieces that had to be addressed. So sometimes it's better to just throw the old stuff away. Sorry, but in my opinion, that's what we've got to do.

Then the other one is frankly we have a zero risk tolerance culture. In the space business, that is true in spades. Now don't get me wrong, I, like all the other SMC commanders between me and 1999, did not want to be the first SMC commander to certify a launch that failed. I bowed to the mission assurance gods three times a day and

made sure that we were ready to go.

However, that risk aversion has caused us to layer checkers on top of checkers and slow down our processes to the extent where we are, and there's no way we can speed up that. So we have to be able to temper our appetite for risk mitigation if we're going to be able to do this. That gets back to empowering two- and three-stars to make decisions about how something should be operated, or even at a lower level than that.

The other thing that we have is this idea that software is developed and then sustained. What the heck does that mean? Remember that thing about this is a continuous process? A number of times I get bogged down in these discussions and I say okay, we're going to this contractor to develop the software. We're going to acquire all the data rights so we can compete the sustainment.

Remember when I told you how many software engineers they hired at John Deere? And Goldman Sachs has 800 of them. They live there, they own that, they have software teams for life.

This isn't okay we developed this thing and now -- how do you sustain software? Software doesn't break. You may find something that doesn't work the way you thought it was, but it doesn't break. You don't bring it in for corrosion mitigation or overhauling the engines. When you look at what we really do in software sustainment, a lot of it is continually improving the software.

So because of that, we also get ourselves bogged down in this color of money, and this is where I think you all can help us. If you go down to Rich Aldridge who runs the business systems -- and if there's anywhere in the Air Force that we should be able to do agile software development, it's in business systems. He will tell you that he ends up constantly being bogged down because someone will decide that the research and development money that needs to be spent to do that software thing that his customer wants, and he doesn't have any research and development money, he only has O&M because that particular system was in sustainment, and we do sustainment in O&M. And by the way, it's a two year process to get into the POM.

If you have RDT&E then you get categorized as some kind of ACAT program. Therefore you now -- remember systems engineering V because you're going to have to have a systems engineering plan, and you're going to have to have all this other stuff. And you may even have to go through a DAB to get the thing approved. And all of a sudden, where did my agility go? All I wanted to do was to make something a little bit better.

I'll give you an example of one I just went through with this. Our aircraft simulators: Key and essential to the future of the Air Force is to be able to do live, virtual, constructive training. First of all, we cannot afford to continue to fly the airplanes and big exercises. And second, with some of our sophisticated weapons and capability we really can do that more effectively by doing it with virtual and simulation.

But our simulators were all built separately with proprietary software and code. And by the way, we have this thing called cyber security, and each one of them has cyber vulnerabilities. So we came up with a strategy that puts them into a common architecture, which actually will help us with cyber security as well, and an approach of doing this with agile software development and we would eventually get them all in.

We came up with a strategy for doing this using our sustainment dollars, because we have to address those cyber vulnerabilities, because that's a broken software, right? That's not new capability.

But in the process of doing all this we're going to really improve and enhance this. We're going to improve the flexibility of it. We can actually incorporate some new capabilities.

The lawyers and the FM'ers came to the conclusion that I can use O&M to do this only if when the pilot walks into the simulator he sees no difference. As long as I can get away with that, it's okay, I don't need RDT&E money. Is this absurd or not?

But this is all back to this whole concept. If we go to the next chart -- let me see if I missed any other barriers. I think I hit the highlights.

What do we need to do? First of all, we need to eliminate disruptive organizational barriers. I call this, for lack of a better word, the Goldman Sachs model.

We need to bring the coders and the operators together. I am not saying that this -- the immediate reaction when I told General Goldfein this he goes, do you mean I need software squadrons? I'm like, well yeah, but I don't want you to think about this squadron in the traditional sense where everybody is wearing this kind of uniform.

I think these software teams that will be co-located with the guys that are doing the operations, could be a combination of blue suiters, civilians, and industry teams. I don't care. I want the best people. Remember those self-forming teams? That's the most effective way to do it.

So I am looking at implementing strategies where we actually co-locate and use, to a certain extent, internal software coders. I learned, as I was going through this, we have 3,000 software coders in AFMC. They work in our air logistics centers. The write the operational flight profiles out at Hill [AFB, Utah]. They write code out at Robins [AFB, Ga.] that goes into the DCGS.

So we're already doing this. But those guys are not trained -- they don't do agile software development, which by the way is a problem when I'm trying to hire new people because all the people coming out of our universities only understand agile software development. So I'm having to bring them in and un-train them in agile and train them back to that systems engineering V and all that paperwork.

If you think I have trouble recruiting them, yes. They don't want to be bogged down with all this. The hoodies, as we sometimes say, they don't want to be bogged down with that.

So this is an organizational change. This is thinking about -- and think about all the other barriers. I've got to overcome this independent tester. But I think -- this is where I say 90 percent of that is internal, its' cultural. We have to change it.

We have to look at how we do our contracts to enable us to do that. I hate to say this, but maybe it's time for time and material contracts again, so that I have the agility to figure out -- and I know that's blasphemous when we talk about bad contracting strategies -- but I have to have the agility in my contracts to be responsive as the requirements change.

The second thing is, okay, so I've put the teams -- and the teams are there for life. I don't mean that as one person, but we don't think about -- okay we only put this team together to do the development and then they're out the door. That team stays with that system forever. Maybe that's why, when I put this in terms for someone like General Goldfein, maybe the right thing is to say yes, I'm going to have software squadrons. There are going to be places where there will be an organization that will do that.

We need to make the user the operational tester and acceptance authority. We've got to eliminate this independent tester. That's one that is kind of in that five percent because the DOT&E role is in law.

But we have to be able -- you know, the Defense Digital Service tells the other guys to go out and put the software in there. The guy who wrote the software is leaning over the shoulder of the operator. When something goes wrong he says, I know where that is, because he knows the code.

He didn't have a stack of software documentation that was developed when the contract was delivered. He knows the code. It's his.

We need to get over this dollar thing, this color of money thing. I don't really care how we do this. People say, well, let's just do it all with O&M. That becomes the trust issue because the issue is one of them has the least restrictions on it in terms of it, and how do we trust you? That's what the FM'ers tell me. That's what the staffers tell me. How do we trust you?

So I don't care. If you don't trust me with O&M then maybe we need software production money, like we have aircraft production money and we have set space satellite production money. In the Air Force vernacular we have 3010, 3020. Maybe we need 3090 where it's software production and we do it with a separate color of money.

But we have to overcome that barrier. We have to reframe that systems

engineering process so that it isn't as hardware centric. And we have to transition -- this is back to me -- the task. Remember what started me on all this was when my boss told me we need a net centric approach to managing systems instead of stovepiped weapons systems.

This gets back to how we organize to do acquisition. I have a B-2 SPO. I have an F-22 SPO. So when poor Linda Rutledge is trying to figure out how to pull these simulators together, she has to go to all these different entities.

That's a deeper one for me to contemplate and think about, but how do we eliminate -- if we're really going to be network centric in how we think about our war fighting capability, I think we're going to have to think a little bit about how we in the Air Force define a weapons system and how we organize to support it. Right now, the network is the tail on the dog, and the weapons systems are the ones that call the shots.

Just think about the challenges we've always had in getting new GPS capability on our platforms. Well, we only do it when it fits into the schedule for the hardware. That's not going to work, right? So that's where we are there.

So how are we going to do this? Some of it is cultural. Some of it we're going to have to drive changes in DOD and Air Force policy. I've been pushing to try to get a couple Pathfinders and pilots going so that we can demonstrate this.

And I think there are some legislative things that we can do that will help with these barriers. But I don't see any of these as insurmountable. More importantly, I see them as critical if we are going to be able to continue to provide the kind of war fighting capability, with the agility and the flexibility that we're going to need, for this country.

This is probably the first time you've heard me talk about this, but you're going to hear me on this again and again and again, because this is so critically important. And it's not just for space, frankly it's across the whole Department of Defense. It's not just for space.

With that, I will open it up to questions.

(Applause).

MR. HUESSY: Tony, do you have a question?

GEN. PAWLIKOWSKI: Tony, you don't have a question?

MR. HUESSY: He'll make one up.

MR. TONY CAPACCIO: The Air Force yesterday canceled the Northrop AOC contract because of software issues. How does that fit into your notion of problems that need to be solved?

GEN. PAWLIKOWSKI: That program, that capability, is at the heart of what we're talking about here. That is actually one of the -- I don't even want to call it a program. That capability that we're trying to get into the AOC that 10.2 represented, is one of the key drivers to tackling these issues. Under Darlene Costello [SAF/AQ] there is a Pathfinder that we are kicking off that tries to capture a lot of the things we talked about here.

MR. TONY CAPACCIO: How much does the cancellation throw back your effort?

GEN. PAWLIKOWSKI: Not at all, not at all. Remember, the requirements are still there, right? So I have the opportunity now to try to get after those requirements using an agile software development construct, as opposed to a traditional -- okay, I have this big ACAT-1 program here that's going to cost this bazillion dollars that's going to take this amount of time and I can't start it until I have a DAB and get approval through the POM. We're going to start to get after those requirements using an agile software development construct even as we're speaking here.

MR. TONY CAPACCIO: You spent half a billion dollars and the last four or five years on this that you're just going to lose?

GEN. PAWLIKOWSKI: Well, there are elements of it that we will be able to use as we go forward, particularly some of the work that has gone on in the last time period. Tony, I don't have the details as to exactly what all the money was spent on, but not all of it is going to be completely thrown away. We'll be able to use parts of it, particularly the architecture that was started -- the work that was started on, as we go forward.

MR. HUESSY: A question here.

MR. WILSON BRISSETT: Wilson Brissett from Air Force Magazine. General, do you think the formation of a separate Space Corps could help this process development, streamline the acquisition, make things faster in the way you say they need to be?

GEN. PAWLIKOWSKI: What we're trying to do here has to be done regardless of what the overall organization structure is. I don't have any -- I frankly think that we can do what we need to do and work within whatever construct we decide as a nation is the best way to go when it comes to that. Our chief and secretary have laid out and been very clear about where they think we should be going.

My focus has been to try to make sure that the acquisition community can provide that agility that needed to provide the capabilities we need in space. The things I talked about here, a lot of this is at the tactical acquisition level, when you think about it. So I'm very comfortable with going forward under the current construct with this and we'll adapt and adjust as decisions are made.

MS. VALERIE INSINNA: Nice to see you, general. I had a question also about AOC-10.2. I was wondering if you could provide some information about why the Air Force decided to cancel Northrop's contract? I'm asking just because OCS was running into problems and you guys started doing a DevOps software development. For that, you guys got Raytheon on board. So I was wondering, what was the difference between those two programs.

GEN. PAWLIKOWSKI: Well, they were at different phases in where they were. The assessment that was made, when we looked at both programs, was that the best opportunity to get the capability in OCX was to stay the course. In fact, in the OCX program today there has been active involvement by help, if you will, from the Defense Digital Service in applying some of the techniques that we've talked about here. The decision in the case of AOC was to take a different approach than to stay the course with that one.

MS. VALERIE INSINNA: Can you talk a little bit about what your work with industry on this program is going to look like going forward?

GEN. PAWLIKOWSKI: The AOC one?

MS. VALERIE INSINNA: Yes.

GEN. PAWLIKOWSKI: I really can't. I am not involved directly in that on a daily basis. I think it's probably better for Steve Wert [PEO Battle Management] or Ms. Costello to answer that question.

(CONGRESSIONAL STAFF MEMBER): General, thanks for coming. As an amorphous Hill staffer there's nothing I love to talk about more than agile software development.

(Laughter).

GEN. PAWLIKOWSKI: A pretty boring topic.

(CONGRESSIONAL STAFF MEMBER): We've seen the successes and the progress DevOps had with OCX, so there is certainly a business case in line to support what you're arguing here for. In a different program -- and I'll leave the organization and the program out of it -- they're trying to implement agile software development. They came to brief the Congressional staff on, here's our progress and schedule on the program.

Where's work flow chart? That's how we do our oversight. That's how we communicate to members if things are going well or not.

We could see on the schedule we had some problems. There's no milestones, no

nothing, but one year from now we're going to be doing okay. It was sort of that trust issue that you brought up.

I think you've got the business case behind you that says Congress, somebody at the peon staffer level, should be trusting you. However, there's no accountability built into it that we can see. How do we know?

GEN. PAWLIKOWSKI: My first reaction to that -- and I don't know the program, although I can probably guess what organization it is -- is remember what I just talked about, delivering working software. I would actually go back and say, a year is a long time in an agile software development process. There's got to be something in between that they can be able to demonstrate to you that they're delivering. If it is taking them a year before they actually get anything into the hands of the operator, then I guess I've got some concerns about their application of agile software development.

MR. COLE LEE?: From the appropriations side and the squirrely green eye shade FM side, when you talk about color of money I'm climbing --

GEN. PAWLIKOWSKI: I know, I know.

MR. COLE LEE: I see Mr. Rogers and he's yeah, go, go.

(Laughter).

MR. COLE LEE: I hear you. But I would encourage your staff that as they make these proposals and these Pathfinders to sort of build in an -- okay, we might need to muck around on color of money. We might need to do this. But here's where the new accountability is. Here's how we think that oversight can play, so that our members can be confident in this brave new world that started 12 years ago in the Department of Defense and which the Congress is just catching up to can stay on top of.

GEN. PAWLIKOWSKI: I agree with you. There's a lot of things -- like I said, there's some things you have to think about. That one, I've thought about that one. How do you -- it's like I said. I'm not comfortable saying just throw O&M money at it.

You've got to have some way of understanding that you're actually getting what you think you paid for. I again go back to those basic tenets of success that it's working software in the hands of the operator. So in my view, the key is being able to look at those requirements and, if you will, break them up in a way that you are incrementally fielding capability in increments that are significantly smaller, maybe, than the way we've thought about this in the past.

You know, incremental acquisition is not new, right? We've talked about that. But we've always talked about -- I mean, GPS-3 originally started with an A, B and C write out, and we never got past A. But A was like a four year program.

So when we talk about agile software development, A is probably maybe six months. So that's where I worry a little bit about that timeline. I'd say we have to -- and that's one of the discussions that I have talked about. What's your metrics? How do you measure that you're successful?

But I keep going back to those basic tenets. My measure of success should be the amount of capability that I have delivered in an amount of time with the quality and satisfaction by the customer. So how do we do that without requiring an independent tester to write a report that takes them four months after they've done their testing, and then want to make a decision to accept it? We've got to be able to capture the quality and the effectiveness of the product we deliver so that we can communicate to you.

Even authorizers like to be sure that we're spending the money on what we said we are and we're getting a product from it, right? So I think that the key part of working through this is to build that trust, but to have a way to measure our effectiveness that doesn't create a huge volume of documentation and delays in order to communicate it, if that makes any sense?

I don't have all the answers. But my view is if I don't start down this path, it's kind of like agile software development, let's give it a try. We may have to circle back and say that didn't work. But if you fail early, which is the TomTom approach, then you have more time and you haven't spent as much money getting there.

MS. (INDUSTRY REP): I think the vision that you outlined is really, really quite ambitious and really (impact ?). I think the examples that you've given, like OCX, I would call them hybrids because -- (off mic) -- system engineering process. So I'm wondering, the Pathfinder that you mentioned, I'm wondering what you have in mind to get to that ultimate agility?

GEN. PAWLIKOWSKI: I agree with you that if you try to tackle the most difficult challenge, the most complex challenge with a Pathfinder, then you may not learn anything. What I'm trying to do is a Pathfinder that is on a weapons system that is very software intensive. OCX, my argument is, a bit of a Pathfinder even today that is testing some pieces of agile. But I'm looking at a different one and I don't want to talk about the details of what that might be because it's still very much in the storming phase.

And then I'm also looking at one on the business systems side. The business system side, I think, is much -- which is as you heard me earlier say, is one that we ought to be able to do this. Even when you look at the business systems side, there's the risk equation. Not to say that business systems can't be risky when you make a mistake on them.

I have to admit, I myself am a little nervous about doing agile software development even on an operational flight profile, and finding out I did something wrong by losing an airplane. In the space business, the software for the satellite, same thing. So we have to have a little bit of a calibration when we say accepting risk, because for us the

consequences of accepting risk can be very, very extreme. So that's the part that I personally, as an engineer, struggle with when it comes to that.

But the one Pathfinder will be a business system. That's one I'm willing to share because it's a little further along. What I'm looking at there is to help myself. Every once in a while it's a good thing to be your own customer, right, which is logistics systems. We have over 200-plus logistics systems to operate our Air Force.

And by the way, I have to be audit ready. To be audit ready I have to be able to account for where every dollar is and where it went. Think about the nightmare of that with 200 different business systems associated with logistics.

So one of my Pathfinders will be essentially an in-house effort, using some of those 3,000 software coders we have already, to work on logistics command and control. Again, there I will be able to break -- I have a little ability to try to answer some of the questions that Colin had. How do I know I've been successful? How big is too big in terms of an increment or a scrum or whatever the latest phraseology you want to use?

I agree with you that it's a challenge. But I put it to you because sometimes we're going to have to think about this from the beginning. Traditionally we've built all the hardware and the money goes to the hardware, and then we're like we've got to do the software now: SBIRS, GPS.

That's where some of this is going to have to start. The Pathfinders are going to be smaller things that enable us to then learn, how do I do some of these things in our risk environment, in our construct, so that I can then expand it to the bigger ones?

MR. HUESSY: With that, General Pawlikowski, I want to thank you very much for coming here.

GEN. PAWLIKOWSKI: Thanks a lot.

(Applause).

MR. HUESSY: We are also doing a post session with the general over here in the Bolton room. If you have any questions about whether you're a part of that group, please let us know. I want to thank you, Ellen, for this.

Congressman Rogers, thank you for all you do. You've put missile defense in the right direction. You've put nuclear deterrence in the right direction. We're going in the right direction in space.

I remember when we talked a couple of years ago you said those were your three goals as chairman, to get us going in the right direction. You're moving well in that direction. A lot of us in this room, that's what we do for a living and we want to thank you. Really America's security rests on that, and thank you congressman.

(Applause).

Thank you all for being here and we'll see you next week with Ambassador Ron Lehman from Lawrence Livermore. Colin Lee, thank you for coming.