25 August 2005, Keynote 8:30AM – 9:30AM

[Video playing]

[Applause]

Justin Rattner: Thanks Steve. Thanks everybody. I have to admit that when I first heard about Radio-Free Intel, I was taken by surprise, and I said, "Boy, we've really done it now. We've got our fannies hung out a little too far. But I'm here with Soumyanath Krishnamurthy, who is head of our Communications Circuits Lab. And he's the one actually in charge of developing the technologies in support of the Radio-Free Intel concept. So, Soumya, where are we with Radio-Free Intel?

Soumyanath Krishnamurthy: Well Justin, I'm glad to announce that we've made significant progress. Let's start by looking at what a typical radio model looks like today. You've got a board with an antenna and a few front-end components. You've got a radio and you've got a baseband chip. And if you look at how the radio works, the signal comes from the antenna. It's a very small signal, as you can see, its a few micro-volts. You've got to send it out to the passive components near the antenna. You've got to send it to an amplifier and gain it up. You’ve got to put it to a mixer to change the frequency, filter it, and then convert to digital with the A-D converter.

And on the rear side, the same thing happens backwards. You start with the bits, you go through the same sequence of events, you've got to pump up your signal by the time you get to the amplifier, and then you send it out to the antenna and it goes out to the airwaves.

Lots of people have tried to integrate this, and they've done a reasonable job of integrating the middle section, but we've gone a whole step further. We've taken the antenna, the front-end components, and the middle section, and collapsed it together on one little die. And that's the die shot right there. And you can see the structures there. There are the structures on the antenna that show the matching networks for the low-noise amplifier, the power amplifier, and it's built for 2.4 and 5 GHz. And that's what we've managed to accomplish in the short time, and that die is our Intel die. It's flip-chipped onto the package, and you can see the various elements on the package that shows the routing and the matching filters all in one little thing. And here it is, Justin.

Justin Rattner: Amazing. Wow, it's really tiny. Look at that. Well, that's the beginning of Radio-Free Intel, I guess. So, you know, these communication standards are always changing, and you know, bandwidth grows – is this going to be able to respond to future standards?

Soumyanath Krishnamurthy: Excellent question, Justin. Absolutely. The die we have here today, this is ready for 100 MHz baseband bandwidth. Today's radio’s only require 20 MHz. So we're ready to basically deal with anything that a new standard like 802.11n can throw at us. So we're ready for the future.

Justin Rattner: So sometimes digital technology gets us in trouble, from a power point of view. How does this do in terms of power efficiency?

Soumyanath Krishnamurthy: It's great. Moore's Law to the rescue again. We're totally dependent on Moore's Law. This thing is built on 90 nanometers CMOS processes, our workhouse digital process. We use the low voltage device throughout. It's a 1.4 volt supply radio; it's very low power – 140 to 150 milliwatts, its industry-leading power envelope. So all I can say is its flexible, and thank God for Moore's Law.

Justin Rattner: Okay, but that's not the whole radio. This is basically the front end. How are we doing getting the whole thing down to single chip?

Soumyanath Krishnamurthy: Okay, so this is basically what we'd like to do is to start at the antenna – this is the holy grail of the future of silicon radio – we'd like to start at the antenna, collapse the entire middle – it just vanishes. We go from antenna to bits in one fell swoop. And that's the vision of the future.

Justin Rattner: Wow. A radio designer's dream.

Soumyanath Krishnamurthy: Exactly.

Justin Rattner: Thanks very much.

Soumyanath Krishnamurthy: Thanks.

Justin Rattner: We appreciate your coming today.

Soumyanath Krishnamurthy: Thank you, Justin.

[Applause.]

Justin Rattner: Well, Steve, it looks like we're doing pretty well on the Radio-Free Intel vision. Do you have any other things you'd like us to work on?

Steve: This is kind of getting a little bit embarrassing. Why don't you go ahead and give your keynote, and I'll try to think up something really juicy?

Justin Rattner: Okay. We'll get back to you later. Well, good morning. Let me add my welcome to the third and final day of fall IDF 2005. It's great to be here again. I promised everyone I'd come back in the fall and extend this vision of the future platform, what we call Platform 2015. I really feel it's my role at IDF to get you in front of the technology. Steve's job is to look behind the technology, but we want to keep you on the leading edge. We want you thinking about what platforms can and should be five or ten years into the future.

This morning we're going to focus on this notion of user-aware platforms. Before I get into the formal definition of what I mean by user-aware platform, I thought it'd be useful to look at some other intelligent platforms. For those of you who are sci-fi buffs, you'll find this little tour quite familiar. Not only were the platforms in these science fiction movies user-aware, they were self-aware, and that notion of self-awareness added to the excitement of those motion pictures, and actually got them into quite a bit of trouble as most of you know.

One of the most famous of these self-aware platforms was HAL from 2001: A Space Odyssey. HAL was just amazing, totally connected. He could actually feel what the ship was doing; "I'm feeling good, the ship is feeling good, everything's good." But in the end HAL turned out to be too human-like. That was a fatal flaw. When he learned that the crew planned to turn him off, then he started killing the crew, and in fact nearly killed all of the crewmembers.

So you can see this concept of self-awareness can be a bit dangerous. Here's another example from Terminator. It's the year 2029 and something called Skynet – I guess that's sort of like the Internet only better – comes online, and it's equipped with the dreaded neural processors. I don't know what they are, but they're dreaded anyway. And the key thing is that neural processing environment begins to learn and do so at a geometric rate. And of course Skynet becomes self-aware – it's beginning to sound familiar – self-aware and soon attempts to destroy the human race. And three movies later, it's still attempting to destroy the human race.

In both of these views of the future – both 2001 and Terminator – this notion of turning against the user is fundamental to the plotline. But it doesn't have to be that way. In fact there's a classic science fiction film that I think really represents the ultimate in what I'm calling user-aware systems, and that's Forbidden Planet, which featured this wonderful robot named Robby. Good morning, Robby, how you doing?

Robby: That is correct, sir. For your convenience, I am programmed to respond to the name, Robby.

Justin Rattner: Okay, Robby. You know, you're in pretty good shape for a 50-year-old robot.

Robby: Krell metal does not wear out, commander.

Justin Rattner: Oh, that's right. The Krell. Well, why do you think Robby was so user oriented? Why is he so different than the other robots in these science fiction movies?

Robby: I incorporate superior Krell technology, commander.

Justin Rattner: Oh, that's right. Krell technology. A million years advanced of human technology. Well, you're going to think what we're showing today is fairly primitive.

Robby: Intel is about as close as you can get to Krell science, sir.

Justin Rattner: [Laughs] Well, I’ll tell my researchers back at Intel what you said. I’m really glad you’re here today, and I have a million more questions for you. But, I’ve got a keynote to finish. So, if you’ll just hang around a bit, I’ll talk to you after the show.

Robby: A genuine privilege, commander.

Justin Rattner: Thanks, Robby. Well, Robby was really a wonderful friend, a selflessly obedient servant, and most importantly, he had built-in safeguards. He cared more about his users than he did about himself and was willing to give his life on behalf of his users. Unfortunately, today’s systems are some distance from the sophistication you see in a robot like Robby.

Today, as most of you know, we really direct every move in the various platforms we deal with. We have to be very explicit. It’s a very tedious process; if you don’t type the filename just right, you can’t find the file. It really doesn’t matter whether you’re just doing routine word processing or presentation development. It’s also true when you’re managing battery life. I’m constantly reconfiguring the mode of my notebook to tell it, “Hey, I’m here, I’m there. I’m using the DVD player; I’m not using the DVD player.” So, I spend a lot of my time really worrying about sort of the care and feeding of the platform. Then, in the worst case, if I suffer a virus attack, I spend a lot of time cleaning up the system, getting rid of the virus, making sure I didn’t lose any files. We really need to get to a point where our systems support the user rather than users supporting the system; in other words, we need to make them user-aware.

What does it mean to be user aware? What’s that concept all about? Well, I define a user-aware platform as one that’s able to take care of itself. I remember Robby giving himself an oil change in Forbidden Planet – a pretty useful thing for a robot to be able to do. A user-aware platform also has to be able to conserve its resources. It’s got power systems and whatnot, and it has to make sure that it’s using its energy systems and resources most efficiently. It needs to anticipate what we want done. It needs to be thinking ahead in some sense and planning for the next action on the part of the user. Of course, it needs to be able to sense its environment. How can it understand what our needs are if it really doesn't understand the environment in which we and it exist in? Finally, and perhaps most importantly, it needs to learn and adapt to our needs, and that ability to learn and adapt I think is fundamental to this notion of user-aware systems.

When we achieve this level of sophistication, I think we will fully unleash the power of the platforms we build and achieve the ultimate in simplicity. That will be the case whether we're talking about, whether we're talking about handheld, notebook, desktop, or servers platforms – or maybe platforms like Robby that have yet to be invented.

To get a better sense of user-aware systems, I want to start with a fairly familiar example, that of taking digital pictures. I meant to bring my digital camera and take a picture of you guys. If you're like me, you take lots of photos – hundreds a month. I understand some people take thousands of digital photos a month. I can't even imagine doing that, but I certainly take hundreds of them a month. I dutifully load them onto my hard drive and then forget about them. Yes, sometimes my wife will say, "Gee, do you have a picture from the wedding?' or whatever it was, and off I go trying to find it. Inevitably, it takes me way too long, and I get frustrated. In some sense, this is a broken user experience. Digital photography is wonderful, but we've got a crisis in the making. There are going to be tens of thousands of pictures spinning around on hard drives all around the planet, and most people won't be able to find a picture when they need it. My wife has complained she'd rather just go back to her shoe boxes. Of course, I told her she'd need a really big shoe box for all of those pictures.

So, we need a simple way to search for digital photos, one that's natural and familiar. We need to be able to search like we think. Let me show you an example. Please welcome, Rahul Sukthankar, who's a Principal Research Scientist at our Pittsburgh research lab. Come on in, Rahul.

Rahul Sukthankar: Hi, Justin. Very good to see you again.

Justin Rattner: Good to see you, this morning. I understand you're also associated with the robotics lab at CMU.

Rahul Sukthankar: I am, that's correct, with the robotics institute.

Justin Rattner: You cooking anything up like that? [points to Robby the Robot]

Rahul Sukthankar: We are, but we're not ready to show it this year.

Justin Rattner: Okay, a future idea.

Rahul Sukthankar: Absolutely.

Justin Rattner: That will be great. Well, you know, Robby might need an upgrade.

Rahul Sukthankar: I'll take him back with me –

Justin Rattner: No, no, he's staying here. Okay, so I've got this problem – I've got all these digital pictures and they're just hard to find when you need to get one right away. Are you guys working on anything in Pittsburgh?

Rahul Sukthankar: Sure, let me show you a little bit about the Diamond project. So you mentioned shoeboxes. Imagine your computer just contains lots and lots of unorganized photos. In this demo, we have 85,000 photos just tossed in there. Our goal is to look for a particular photo or set of photos. So for example we're looking for a photo of yourself from one of your previous presentations. How would we find it if it was completely unlabeled? It forces you to do some kind of brute force search.

To give you an idea of what that would be, if you were just to look randomly through these photos, it would take you hours if not days to look through these photos and look for something specific. Now people have been working in computer vision and machine learning to understand content inside photos, but it's still fairly crude, so if we could take the analogy of looking for needles in a haystack, what we want to do is really throw out as much hay as we can.

Justin Rattner: Okay.

Rahul Sukthankar: So let's start out by using a well-known technique such as face detection to whittle down this set of photos, and we'll use a very conservative notion of a face. So anything that looks even remotely like a face will pass through, and anything else we'll just reject right at the storage device before it even gets up to the client.

Justin Rattner: So I'm the kind of person who sees faces in the clouds –

Rahul Sukthankar: Absolutely. So here we're going to be willing to say, we see faces in string beans and clouds and so on, but we're starting to see a few images there that look face-like. You can see it's doing something useful, but it's not actually useful enough at this point to answer our query because it's finding faces but not really finding faces of you. Now I remember when you gave a previous talk, you were wearing a traditional blue Intel shirt. It looks like you're trying to fool me today with the pink one.

Justin Rattner: Yeah, I kind of tricked you here.

Rahul Sukthankar: So what we can do is we could look at an image from a previous IDF – there's an example of someone with a blue shirt –

Justin Rattner: There's Pat.

Rahul Sukthankar: It's Pat, and we could say, okay, well we want to look at colors from a shirt that look kind of like this, so we just give it a few examples of this.

Justin Rattner: So you're actually creating a new filter here on the fly.

Rahul Sukthankar: Absolutely. It's very easy because we're not indexing the data, so it's very easy for us to create new filters in an extensible manner.

Justin Rattner: Okay.

Rahul Sukthankar: So we call this the little blue shirt filter, and now we look for faces and for this blue shirt, hopefully we can refine the search so we find things that are more useful. Now if we look here we can see that it's finding images of people from IDF. We don't have face recognition here, so it can't recognize you, but between us we can see its dropped in more than 99.9% of the images, and then just by looking forward we can say oh, look, there you go. Hey, there's an image that looks like the one we want, and that's an example of Diamond in action.

Justin Rattner: Fantastic. Well this is incredible technology. Where do you go next?

Rahul Sukthankar: So this is a collaborative research project with Carnegie Mellon University and Intel. We're now working with the University of Pittsburgh Medical Center and with Merck Research on biomedical imaging.

Justin Rattner: Well that will really compliment Intel's initiatives in the health sector, so that sounds great. Hey, thanks for coming today.

Rahul Sukthankar: It's my pleasure.

Justin Rattner: We really appreciate that.

Rahul Sukthankar: Thank you again.

Justin Rattner: Thank you. Well I hope you're starting to get an idea for what I mean by user-aware systems. That's the kind of tool I could turn over to my wife and have her zooming through our photo files and efficiently locating the kinds of pictures that she wants. Let's move onto a different aspect of user-awareness, one that I touched on a few moments ago, and that's this notion of being able to have a platform that takes care of itself.

This aspect of user-aware systems would be particularly valuable in big data centers. In that case, the total number of systems that you find in one of these data centers is really enormous, and with Blade-based servers, we could be talking about tens of thousands of discreet server elements. It's so complex that if something fails, it's almost beyond human capability to identify the failure and respond to the failure before there's either the loss of data or the physical destruction of the system. So we need to create technology which in fact is able to deal with that kind of scenario.

Here we have an example. This is a typical mid-size computer room. The thermograph, by the way, is courtesy of Hewlett Packard's smart data center program, and we thank them for making it available to us. Just to orient you here, we've got rows of server racks moving up and to the left, and we also have a bank of air conditioning units here at the bottom. We'll assume that the data center is well planned and adequately provisioned with air conditioning. So in normal operation things look pretty good.

But should an air conditioning unit fail, things really get out of hand in short order. The temperature in the room starts to rise and now we have to act very quickly or we may lose data -- or worse we may destroy one of those servers, or entire banks of servers. This is not the kind of situation that a human administrator is going to be able to respond to in time to prevent either of those two occurrences.

What we need in user-aware systems is the ability of those platforms to deal with this kind of situation on their own; to handle the change in the environmental condition and take the appropriate action. If you think about this, it's much like the autonomic systems that maintain our body temperature and regulate our other systems. That autonomic nervous system is what replaces having to have someone spray you with water when you get too hot, or put a coat on you when you get too cold. In fact IBM coined the term 'autonomic computing' to describe platforms with these and a number of other characteristics. And I thought who better to explain this notion of autonomics than IBM. I'd like you to welcome Alan Ganek, CTO of Tivoli software and vice president of autonomic computing at IBM. Come on up Alan.

Alan Ganek: Hi Justin. How are you?

Justin Rattner: Mr. Autonomics here with us this morning. It's really a pleasure to have you.

Alan Ganek: Delighted to be here.

Justin Rattner: So what's motivating IBM? What's behind this interest in autonomics?

Alan Ganek: Information technology folks are under tremendous amounts of pressure. Just in the last decade they've gone from environments from where they were managing maybe dozens of servers to hundreds, thousands, or tens of thousands, in many cases. They've gone from managing more servers than they've have endpoints, so the wonderful advances with Moore's law and wavelength division multiplexing and so forth is great for producing technology we can absorb, but that absorption has created enormous complexity. We're at the point now where in many IT shops, typically 80% of the investment is going to maintenance and operations. It's squeezing out the opportunity to introduce new revenue producing applications. So these organizations need some help.

Justin Rattner: What are the key challenges in bringing autonomics to market?

Alan Ganek: Well when you look at the environments, there are information technologies silos; there are different kinds of technologies and different vendors. You've got the hardware; different chips, different servers, different manufacturers, databases, networks. It's a tremendous conglomeration of technology. And unlike the old days when one application from one endpoint went to one server, today a typical transaction goes through dozens of systems, multiple router hops, firewalls, application servers, databases. And in that environment, there's a lack of standards to get any holistic view of what's going on. So these environments need some kind of instrumentation because right now you can't easily figure out what's going end-to-end and where your problem is.

Justin Rattner: How does autonomic computing help that?

Alan Ganek: Well, autonomic is about building intelligence in and the right kinds of instrumentation at every level in the system. Right from down in the chips -- some of the work you're doing at Intel, and that we do at IBM -- up to at the server level, middleware, and right up to the application level. This is about providing the right kinds of standards instrumentation to make that happen, and we're working across the industry. It can’t be a proprietary approach, but an open standards approach, so you can collect all this information and then you can build intelligent behavior on top of that.

Justin Rattner: So is it moving out of the lab? This isn't just blue sky stuff?

Alan Ganek: No, we're taking a very pragmatic approach. This is something we can contribute to, and we've been making really great progress. At IBM we've delivered hundreds of self-managing features and more than 75 products. And we're handling a wide variety of various – take an example like Dynamic Provisioning, where you have widely changing workloads, and you need to be able to move the compute power from where it's needed, from one application to another. A good example of that are the tennis matches, the Grand Slam tennis matches. The websites are red hot when somebody is playing, and then when it starts to rain, nothing happens. Rather than having literally thousands of servers dedicated there, we can move application loads from that to something else, and do it in just a few minutes.

One of the key breakthroughs has been in the area of standards. Working in the Oasis standards body, we've developed the WSDM event format, which describes the standard semantically accurate mechanism for capturing event information, so that you can collect events from hardware, from software, from different applications.

We've put tools together to instrument the environments, and we've gone out and worked with dozens of customers. And on average, we've been able to reduce the time to isolate a problem by 50%. This will be the platform for better automation as we go forward.

Intel's Active Management Technology is an example of extending that kind of instrumentation even further. And when you connect it with our Tivoli software for monitoring, it'll really be able to do great things. For those of you in the audience, at 10:00 a.m. in Room 206 there will be a presentation on Tivoli and AMT together, to take advantage of those kinds of technologies. And there's also a showcase on it down in the Digital Enterprise Zone.

Justin Rattner: Well you know, we've got a room full of developers here, and you can never give them enough information. So if people want to know more, maybe they can't make the classes, where can they look?

Alan Ganek: They can look at http://ibm.com/autonomic. We have a tool kit out there that is available for developers. Intel engineers have taken a look at them, worked on them, and given very positive reviews. It has not only tools, but it has componentry, scenarios, documentation. So it's really trying to get the whole community to get together and take advantage of these technologies, to improve the overall system behavior for customers.

Justin Rattner: Wow, that's great. Well you know, we're doing autonomics research at Intel. Would you like to see one of our latest developments?

Alan Ganek: I'd love to. Terrific.

Justin Rattner: Come on up here. Let me introduce Lenitra Durham. Good to see you. This is Alan Ganek from IBM.

Alan Ganek: How are you?

Lenitra Durham: Nice to meet you.

Justin Rattner: Let's come on over here. Alan, why don't you stand on this side with me? So, it looks like you're cooking something here, Lenitra.

Lenitra Durham: We have a mini data center environment with these two servers. We have wireless Intel motes that are measuring the ambient conditions around each of the servers. And we're also able to get to the integrated sensors on the platform, to find out information about what's happening there. So, on the screens you see we're measuring the CPU load, the internal temperature of the server, each server, the ambient temperature for each server, and the humidity. The top one is the server in the case, and the bottom graph is showing the server that's external.

Justin Rattner: All right, can we heat things up a bit?

Lenitra Durham: Sure. So turn on the teapot and place it inside the enclosure.

Justin Rattner: Do not attempt this at home.

Lenitra Durham: As we wait for the pot to start boiling here and the Intel motes to pick up the changes, I’ll tell you about some of the policies that are currently running. So, the internal temperature – we're trying to manage the internal temperature on these two servers. The server inside the case has a higher ambient temperature, which means the conditions there in that part of the data center may be hot. And you want to make sure that we're not running the server and getting it too much load, so it can't cool itself. The system now detects the change when the steam starts to rise and the humidity increases, as you can see on the red line on the graph. The system then migrates the loads from the server in the case to another external server and actually hibernates the system.

Alan Ganek: So you’ve prevented the damage, but did you lose data in the process?

Lenitra Durham: Actually, no. We migrated the load in time because we’re sensing the conditions, and we hibernate the system so that when you re-power it, it will resume where it left off.

Alan Ganek: Well, I think we’ll be able to take that kind of instrumentation and make good use of it in our monitoring and provisioning products.

Justin Rattner: Okay, sounds great. Lenitra, thanks for coming today.

Lenitra Durham: Thank you.

[Applause]

Justin Rattner: Good job. Hey, Alan, thanks for being with us today. We really appreciate it.

Well, that’s another example of user-aware systems being able to detect changes in the environment. But, data centers are a good example of systems that are operating more on a human time scale. As we saw here, and is true in the real world, we tend to think of our changing needs taking place over a range of minutes or hours or sometimes even over a number of days. But, it turns out if you zoom in to the system, you discover that changes in the platform are actually taking place at a much finer grain. Nowhere is that more important than in terms of the instantaneous power demands of the system.

So, we’ve shifted our view here from human time to machine time. Now, we’re looking at the demand line that’s represented by the internal electronic components of our platform. Here we see the changes are taking place on the scale of microseconds or hundreds of microseconds. There are all these little bursts of demand that are a result of all kinds of things happening internal to the system. It might be the result of a cache miss, having to go out across the bus and cycle the memory. It could be disk accesses, it might be the arrival or departure of network packets; you name it. All of these events are affecting the system’s behavior and especially its power demand at a very fine grain.

Ideally, you’d like to follow that demand line, if you will, with our power supply. We’d like to be able to track that demand. The problem is today’s voltage regulators really can’t respond quickly enough to these almost microscopic events that are taking place in the platform. This results in what we call course-grain power management. As you can see here in the illustration, even though the demand is going up and down in just a matter of tens of microseconds, the power supply is moving fairly slowly, slewing the voltage up and down. It just can't respond in those very, very tiny intervals. As you see, we're losing energy, as illustrated by the red in the diagram. That actually represents a fair fraction of the total power budget in the machine.

We'd love to have a view that looks more like this one where, in fact, we have a very fast voltage regulation capability, and we literally follow that demand line as it goes up and down in time. We squeeze out the red areas, as you can see in the diagram. Now, there's a very small amount of redness left in the picture, and we're doing a really good job tracking supply with demand.

Let me introduce now to another one of our researchers from the Intel laboratories, Paula Thurston. Come on out, Paula. Paula is a hardware architect and one of our leading experts in energy-efficient systems. It looks like you brought me a motherboard here.

Paula Thurston: [Laughs] Hi, Justin. Yes. Here we have a desktop motherboard. You'll notice that the main components here are powered by several voltage regulators.

Justin Rattner: Oh, yeah.

Paula Thurston: This takes up a lot of area and uses a lot of capacitors. So changing the voltage level takes a long time, and oftentimes the CPU is prevented from entering sleep states.

Justin Rattner: I see. Well what can we do?

Paula Thurston: We need to reduce the number of voltage regulators and make them smaller. This will enable us to more quickly change voltage levels and track user-demand.

Justin Rattner: OK, so I've got a lot of voltage regulators here – this one – there's a voltage regulator – and another one – into the wastebasket of obsolete technology.

Paula Thurston: And we've got this one.

Justin Rattner: Yeah, okay. Anything else we can do?

Paula Thurston: While we're at it we might as well integrate the GMCH with the CPU.

Justin Rattner: You're going to take the GMCH off too?

Paula Thurston: Sure.

Justin Rattner: It's not a voltage regulator. I hope you have the solution. Okay. So anything else on here? Oh, yeah. Okay. All right. Well, I've lost quite a bit of voltage regulation capability as well as my graphics capability.

Paula Thurston: Well, you saved space.

Justin Rattner: Yeah, I saved a lot of space and hopefully a lot of power but is it still going to work after I've done all this?

Paula Thurston: Let's take a look at our new research platform here.

Justin Rattner: Okay.

Paula Thurston: You'll notice it's a little interesting here. What makes it special is that for the first time we've integrated the CPU, GMCH, and the CMOS voltage regulator.

Justin Rattner: Okay, I see that right here, and you've got another one in your hands. So this is it. Now CMOS voltage regulation. That's a new idea. Is that going to get us the fast response time?

Paula Thurston: Yeah, this actually operates at a 100 MHz, which allows you to switch the voltage levels within fractions of microseconds. It is also very efficient 85% efficient.

Justin Rattner: Can you give me kind of a bottom-line on that? What does that mean in terms of reduce power consumption?

Paula Thurston: Well, we estimate we can save 15-30% compared to today's notebook without affecting performance.

Justin Rattner: Wow. Well, if my math is right, that means 20, maybe as much as 40 minutes of battery life.

Paula Thurston: I could have used that on the last flight.

Justin Rattner: Yeah, me too. Those flights to Europe take a long time. Well that's fantastic. Thanks for showing us that, and we'll be watching this CMOS voltage regulator technology in the future. Looks like it's going to be very important in these user-aware systems.

[Applause]

Justin Rattner: All right. So there's an example of technology working at a very deep level in the system in order to deliver user value at the top level, and making these systems literally better capable of taking care of themselves and conserving their resources.

Let's look at another example here; another user-aware attribute. This is the notion of knowing where I am. If a system can know where I am, there's a whole variety of services it can deliver to me. And I need to be able to know where I am relative to other devices or where I am relative to other things, resources in the network. Location opens up a whole range of new applications, and there are a number of popular examples of this. Where's the nearest parking location; where can I get a pizza? It may even help me locate a friend. We were supposed to meet here at Powell and Market and I can't seem to find them. If our systems were capable of instantaneously locating one another, that would add to end-user convenience.

But there are other examples of location that require much greater precision. These would include things like asset tracking. Of the thousands of servers we were talking about a few minutes ago, it's very easy to lose one. People are always moving them around at Intel, and pretty soon the IT people can't figure out where a particular server has gone. So we'd like to be able to use precision location to quickly find those important assets.

Now we could use GPS, I'm sure many of you are thinking. But in cities, and particularly indoors, it's very difficult to get a reliable GPS positioning. So we thought another possibility was to use Wi-Fi. You know there are Wi-Fi access points going up every few seconds, and if we knew the position of those access points, and then we could locate from them, we'd have a solution to the precision location problem that is probably more available -- at least in urban areas -- than GPS.

So how do you get precision location from a Wi-Fi access point? Well, one possibility is to use signal strength, as illustrated in the diagram. The notebook is hearing from three access points, and they have different relative signal strength. And we actually studied this approach, and as you can see here in the graph, it is okay, but not that great. It's probably not good enough to give us the kind of precision location we need for many applications. This sort of technology might get you down to 15 or 20 meters. And it’s problematic when you get far from the sources. So if you're really some distance away from the access point, the accuracy falls off considerably -- and you can see that here, illustrated in the graph.

Another approach is to use something like radar, that wonderful World War II technology, where you send out a signal, it reflects off of an object, you time the return, and from the time information you calculate the distance to the target. It's a really simple concept and it's proved tremendously successful with radar. Our approach is similar and it's called 'time of arrival.' Here we send out a packet, the access point receives it, time stamps it, and returns it to the original sender. We calculate the time of flight, and from that we get the distance between the client and the access point. With this approach the accuracy is vastly better. You can see on the graph that it's almost completely precise. No anomalies in terms of time, out to considerable distances here. We're over 70 meters from the access point and still getting precision down to one meter.

So I've got a nice little demo of this technology going over here. You're now in my family room -- sort of. Doesn't really look like my family room. But it will serve our purposes here. I've got my tablet computer here, and the tablet computer is talking to an access point we have a little bit offstage. And it's determining my position in the house. What it's doing with that information is putting the image up on the screen of this plasma monitor that's standing behind me. So it's using location to feed the video to the most desirable display in the area. Now I can move over here. I'll walk over to my den. And now the image is up on the big LCD screen behind me. Again, using precision location it's detected my movement from one room in the house to another room in the house. And I can keep going. Now I'm down the hall. My house has a nice long hall. Sure enough, now the only display that's close to me is the one on the tablet. Now it displays on the tablet.

This is a pretty convenient technology. [Doorbell rings] Oh, somebody's at the door. I wonder who that could be. Hey neighbor.

Doyle: Hey Justin, how are you doing today?

Justin Rattner: Good. Good to see you.

Doyle: Hey, I got a problem over at my place. I have a really strong signal from your access point, but I can't get to any of your movies or surf the web anymore. What's going on?

Justin Rattner: Hey, I'm sorry Doyle, but I installed this new Intel precision location technology and it can tell whether you're inside the house or outside the house, and if you're outside -- no movies.

Doyle: That's it?

Justin Rattner: That's it.

Doyle: No more free ride?

Justin Rattner: No more free ride.

Doyle: How about if I borrow this one? Can I borrow yours?

Justin Rattner: Well, okay, you can use this for awhile.

Doyle: Cool! Thanks! But hold it, this isn't working outside either.

Justin Rattner: [Laughs] Boy will that drive the neighbors crazy. So that's an example of precision location technology. And you can see we were also using it for security purposes, so with location we were able to tell what's inside the house and what's outside the house. That kind of protection actually might be more useful than stronger encryption keys or whatever your technology might have in mind.

Let's turn to yet another aspect of user-aware systems, that's this notion of doing no harm. That was one of the wonderful attributes of Robby the Robot. In the movie you may recall the dramatic scene where the commander hands the blaster to Robby, and Dr. Morbius instructs Robby to shoot the commander. Robby goes into this endless logical loop for fear of violating his prime directive, which is to do no harm. The example we're going to show you today is not nearly as dramatic as that, but I think it has significant financial importance and so it's worth taking a look at.

We all know that computer viruses are really the scourge of our time. In 2004, worms caused an estimated $17 billion worth of damage to computer systems all around the world. The loss in time and productivity, and the effort involved in disinfecting all those systems, was really astonishing. The problem is that worms and viruses propagate so quickly that if you aren't able to respond in a matter of a few minutes, the situation is just completely out of control. Here's an example of the Slammer worm from 2003. It literally flew around the world in about ten minutes. Certainly not enough time for human intervention, and in a lot of cases, not enough time for machine intervention.

Here's another example of the Witty Worm – here you see the number of infected systems versus time, and the graph shows about five minutes of time passage here, and very close to a thousand systems have been infected with that worm in just that short period of time. We've been working on technologies that will help systems police themselves and do a better job not harming the rest of the environment around them. And to show us this technology, I'd like to introduce Dylan Larson, head of our network security initiative.

Dylan Larson: Hi, Justin.

Justin Rattner: 'Morning, Dylan.

Dylan Larson: 'Morning.

Justin Rattner: Glad to have you here.

Dylan Larson: Thank you very much.

Justin Rattner: Well, I understand you've been working on some technology that will help deal with this virus and worm spreading problem.

Dylan Larson: That's right. First we thought we'd illustrate for you just how bad and how quickly these things can spread throughout the enterprise network. What we have here is a network of platforms constructed on a typical enterprise environment topology. What we're going to do is start an infection and show just how rapidly this stuff will spread.

Justin Rattner: Holy cow.

Dylan Larson: Clearly human intervention alone would not help – even if I ran to all these platforms and try and turn it off, we'd have a problem.

Justin Rattner: Wow that was fast. What actually happened?

Dylan Larson: Let me show it to you in slow-motion. You'll see it on the monitor. We've been finding a number of different important characteristics of worms from our research – the number of connections per second that the platform opens up, as well as the state of existing security agents on the platform. You mentioned the Witty Worm earlier. The Witty Worm had a unique characteristic in that the worm actually attacked the tools that were designed to protect it. So the firewall on the platform was actually attacked and turned into an attack vector to wage the attack.

Justin Rattner: Yeah, I had one of those.

Dylan Larson: So you'll start to see, in this example, the connections going through the roof. You see that the software agent that's designed to protect the system has been circumvented, and now it has spread to throughout the network to other systems again, restarting the process.

Justin Rattner: Okay. So what can we do about this?

Dylan Larson: We've been working on the labs on some technology to put into the platform hardware mechanisms to do two things: measure those connections-per-second with a high level of integrity, and then also analyze those host resident protections to make sure those aren't circumvented, or to notify the system once those are.

Justin Rattner: Okay, that's what this board is?

Dylan Larson: Yeah, that's this hardware prototype here. Looking at it from a platform perspective, this compliments software products that reside in the operating system itself.

Justin Rattner: Okay.

Dylan Larson: So shall we plug it in and see how the enhanced system fairs against the worm attack?

Justin Rattner: Yes, but before you get started, let's up the ante a little bit. Are you open to a challenge this morning?

Dylan Larson: I think I'm open to a challenge.

Justin Rattner: Well I thought we needed to introduce a dose of reality.

Dylan Larson: Uh oh.

Justin Rattner: So let me get something out here. Okay, since you're the worm guy, here's a big jar of worms, nice, tasty, delicious, meal worms that the stage hands were kind enough to set out here. How about I challenge you to infect the prototype system, and for every worm that escapes, you eat one of the worms in this jar. Come on, you're a man of technology. You up for this? You can do this! Of course.

[Applause]

Dylan Larson: I really wasn't expecting this. I'm confident that this technology will not spread to the other systems to the network, but eating worms? They’re nasty.

Justin Rattner: I'll tell you what. If you don't let any escape, you don't have to eat any of them, and I'll buy you lunch across the street at the Metrione.

Dylan Larson: Okay, but can we eat somewhere besides the Metrione?

Justin Rattner: Oh, you want something fancier? I thought you liked Sushi? There's a great Sushi restaurant.

Dylan: I do. Let’s do it. So, let's plug in the platform, and let's start the attack. Let's hope I'm not eating worms. In this system, you're actually going to see this device has rapidly quarantined itself. Up above, you'll see again – it happens very, very quickly – but you'll see an initial set of silicon, what we call "the manageability engine," up there, which actively monitors the integrity of that agent, and then also analyzing the network connections per second. Over here, we can see that we haven't broken out into the network, and it doesn't look like I'm eating any –

Justin Rattner: Now you've broken the connection, so there's no way for the worm to escape this system.

Dylan Larson: The platform is isolated, which is good for IT, because IT gets to now minimize the spread throughout the network.

Justin Rattner: That’s fantastic. But can you actually deal with worms that the system has never seen before?

Dylan Larson: We sure can. What we're doing in the labs is we're looking at a number of different heuristics, based on connection states, changes in those connection states, and then the state of integrity associated with those agents running within the platform.

Justin Rattner: Okay, well, I guess the worms will live to see another day. And I owe you lunch. Thanks, Dylan.

[Applause]

Justin Rattner: What he didn't know is we had tinkered with the worm database last night. We were hoping to get one to come out. I thought it would be cool if he chewed on one at least. Anyway, we think this is really exciting research. I don't use the word "breakthrough" lightly, but if we can create systems with this kind of a feature, the ability to do no harm here, in the sense of not spreading a worm or a virus to other systems, I think the benefit to the users will be enormous.

Some other aspects of this research that are important is the fact that it's basically free of false positives. We've run over 8,000 hours of worm attacks on this system, and we had absolutely no false positives and no escapes. It detected every known worm, as well as some synthetic worms we created just to stress the limits of the system. So, that's really what we mean by doing no harm -- not only protecting the rest of the systems around it, but maintaining the integrity of the system itself.

Well, today I've talked about making things easier to use, simpler, more intuitive, in spite of what we expect to be tremendous sophistication internal to the system. We want to be able to design platforms that can anticipate and respond to the ever changing needs of the users, whether they're happening on the scale of minutes or hours, or at the fine grain, in terms of nanoseconds and microseconds. And we really believe that it's the collective opportunity of the community here today to deliver user-aware systems across the entire range of user needs, from the smallest systems to the largest systems in the data centers of tomorrow. And I really want to take this opportunity to encourage all of you to join us in this pursuit.

Well now I should probably turn it back to Steve. Steve, what do you think?

Steve: That's pretty interesting, Justin. Let me just make sure I got this straight. In the future, I'll be able to easily find my pictures in videos, my systems will take care of themselves and they'll know where they are. And, if my computer catches a cold or a flu, it won’t infect the entire galaxy. [Laughs] That’s great. But, I was really hoping to see the guy eat the worm.

Male voice: Yeah, I hoped so, too. Well, thanks everybody. It was a pleasure talking to you today, and enjoy the rest of IDF.

[Applause]

Justin Rattner: Now I’ll go spend some time with Robby.

[Music plays]