IBM

Moderator: Angelique Matheny
January 24, 2008
12:00 pm CT

Operator: Good afternoon. My name is (Robin) and I will be your conference operator today.

At this time I would like to welcome everyone to the Testing Best Practices with IBM Rational conference call. All lines have been placed on mute to reduce any background noise.

After the speakers remarks there will be a question and answer session. If you would like to ask a question during this time simply press star then the number on your telephone keypad. If you would like to withdraw your question press the pound key.

Thank you. Ms. Matheny you may begin your conference.

Angelique Matheny: Thank you (Robin). Hello everyone and welcome to this Rational talks view teleconference Testing Best Practices with IBM Rational. I’m Angelique Matheny with IBM Rational and I’ll be your host for today’s call.

And about today's session, security vulnerabilities in software applications are simply another quality issue, yet how they manifest themselves can result in significant risk to the organization. Testing applications for quality and security is the cornerstone of any compliance management strategy.

And here to talk about best practices and strategies for software testing to meet compliance and regulatory requirements are two excellent panelists, Brian Bryson and Danny Allan. Brian is a Technical Evangelist for IBM Rational Quality Management and Rational brand is a member of the quality management team that plans and delivers IBM Rational software quality toolset. Prior to joining the Rational group in 1995 Brian spent several years in quality assurance holding various positions from tester to QA manager Brian studied and developed the best practices required to implement effective quality automation.

Danny is Director of Security Research with Watchfire, an IBM company. Danny joined Watchfire in 2000 bringing with him several years of business and technology related experience including penetration testing and internal system remediation for one of Canada's largest universities. In his role of Security Researcher he is closely involved with enterprise global customer deployment, researching and evaluating technologies and helping to find and recommend strategic directions for Watchfire's security solutions.

Now you won't find any slides for this teleconference. We want to keep it fast moving and very interactive, so after a quick presentation Brian and Danny will open up the call for a Q&A session. These calls are really for you to get your questions answered by our experts. So write those questions and as you think of them and jump in.

So let's get started. I'll now turn it over to Brian and Danny.

Brian Bryson: Thanks Angelique and good morning and good afternoon everyone. My name is Brian Bryson; I'm one of the Technology Evangelists with IBM Rational. This is a, I think for both Danny and myself, this is a new experience for us with these teleconferences. We've, not only have we thrown out the slides as Angelique mentioned, we've also kind of for better and for worse thrown out the script. We intend them and really hope for this to be an interactive session, and you know we've instructed the call administrators to interrupt at any time if any of you on the phone have an idea or a thought or a question that you know, comes to mind or pops up that you want to interject with.

So a simple star 1 on your telephone will connect you to the operator and put you in the queue and let us know there's a question there to pop in with and Angelique will see if she can get us to stop talking for a moment, long enough to get two cents and address the questions or the issues or the ideas as they come up.

So you know, we do have a few bullet points that we, you know, we assume are going to come up and one of the things that I wanted to talk about is, you know in my role here at IBM rational, maybe I'll put it this way, if you would have asked me a year ago what my role was I would have described myself as a tester effectively. However if you were to ask me that, you know, how that role has evolved and changed today I would tell you that I work with the quality management group, and that evolution from test to quality management is, as much as it's a title thing, is really a spirit that we have started to whole-heartedly embody at IBM Rational.

And I'll tell you why, if you think about testing it really does very little to impact or affect the quality of an application. There's no argument that testing will tell you or give you a metric indicating the quality of an application, you know, if you run 500 tests on your application then, you know, 490 of them fail, I think it's a pretty clear indication that you have some quality issues but more testing doesn't really equal better quality.

Things that impact quality are the way you build the software, you know, whether you manage your requirements or how you track your defects or what, you know, what processes you have in place for development, are you an agile group, or you a iterative team, are you calling more of a waterfall approach? All of those really impact quality much more so than testing, testing really will, you know, give you a metric to measure quality.

So we've really focused on, you know, evolving from testing to quality management and what we're trying to do is address the business need really that all, presumably everyone on the phone, you know, regardless of role, whether you're a tester, whether you're a developer, whether you're a project manager or IT manager, the challenge that we all sort of face collectively as a team, which is really balancing, you know, the balancing act of balancing features, resources and schedules in order to deliver applications that are fit for use and on time and on budget.

It's really that focus on process predictability as a team that we're really trying to impact and most certainly that involves, you know, what many people would call testing tools or functional testing or performance testing, but it really goes much deeper than that. It involves, you know, process automation, you know, the management of defects throughout and enhancement requests or change requests, perhaps more generically across the process. It involves, you know, impacting the process to get all types of testing done earlier, not just functional regression tests, you know, functional testing or unit testing that many people will think of from a quality perspective. But things like what we're talking about here today like security testing, trying to get that earlier into the process to improve the quality, not just report on the quality.

Now our strategy for quality management is built on three pillars and that’s continuous, automated and government. We really want our quality management strategy is in a nutshell is a continuous process powered by automation to govern software delivery.

So continuous means, you know, we’re always at it at every stage of the life cycle, be it functional testing, be it security testing. It is something that we are trying to interject into every rule from developer to tester to security analyst to deployment. Now all of that is something that if you try to do manually you’re not going to achieve, you’re not going to have the effectiveness you need, so that’s where the automated portion comes in. We have the tools to automate the process to make it continuous. And if you get to the point where you have a continuous process that is powered by automation, then you are able to finally govern software delivery and get you to the point where you are delivering quality applications that are fit for use on time and on budget.

And that in a nutshell is our quality management strategy. Continuous process powered by automation to govern software delivery. And that is a big, big change from where we were even just a year ago, when if you had talked to me or anyone really talking about any of our tools on the Rational side we would have, you know, we would have called ourselves a testing company. And as we expanded from testing to quality and focused more on, you know the business of software delivery, we realized that there were some holes in our story.

And one of the big issues that you know, customers were coming to us with was the security side, the risk side. What were we doing to integrate security testing and risk management really into our process? And for a long time we didn't have a good answer. We could point to external parties and, you know, we would point quite often to Watchfire the security company and say, you know, here's an excellent company that we recommend and we have some APIs and they have some APIs and you can glue it all together and all the rest of it, but we really didn't have an integrated security store, we really, it was a missing piece of the puzzle.

So about, in fact, I'm just looking at the calendar, roughly a year ago we integrated the Watchfire organization into the Rational brand and, you know, we proudly now have a much more complete perspective on quality management that includes, you know, our traditional strongholds of functional and performance testing and some of our unit testing products and services, but we now have that very important and missing security piece that really addresses some of the most common risks that are out there, especially when the majority of our customers are developing Web applications.

So you know, we have the tools but we're lucky today that we actually have the live representation and sitting virtually with me today is our security research director, Danny Allan, from the Watchfire organization that's now part of the IBM Rational brand and maybe Danny I'll hand it over here to you at this point, you know before we start jumping into any of the questions to talk about maybe you know, some of the outcomes or the risks of security vulnerabilities and just some of the business impacts of, you know, of an incomplete security testing plan.

Danny Allan: Thank you very much Brian. This is Danny, I'm happy to be with you and yes, let's, I can begin here by talking about some of the potential outcomes that I have seen. Now I'm going to start with kind of technical outcomes, but what I would like to drive home as much as Brian was speaking about that this is a business issue. The business outcomes are actually much more severe than the technical outcomes.

But let's start with the technical outcomes so that we can understand the business outcomes. It's important to recognize first of all that there is no such thing as a secure application nor is there likely to be at least in the immediate future. So security is something that we would like to attain too, but it's something that we will probably never ever completely inherit. What you want to look at is the risk that the organization faces or that the client faces because it's important to recognize that there's two constituents when you're developing software. There's the organization or the server or the hosting environment and there's the client on the other end of that.

Now we typically focus on outcomes that effect the organization or that effect the server environment and we do that naturally because that is the piece that we typically own. And so when we look at the technical issues that could face an organization that is hosting in secure applications or in secure software, certainly what we read about the most is probably two primary issues and that is infrastructure control, someone can remotely come in and control the infrastructure, the assets that I own, and then as kind of a spin off from that but equally damaging most of the time to most organizations is data theft.

So we hear all the time about someone who broke in and stole X number of thousands of credit card numbers or some has broke in and stolen personal information and Social Security numbers, that type of thing. So certainly the infrastructure control and the data theft are the two things that most organizations focus on almost immediately because their brand is at risk and it is something that's very tangible and easy for them to recognize as a risk.

Then going down the totem pole, and that really depends on the type of organization you are; some people are concerned about denial of service. If you are a large e-commerce environment and your software and application infrastructure went down for a period of time, even if it's only five minutes that could represent tens of thousands or hundreds of thousands of dollars. So a denial of service where the environment is not available is also on that kind of risk factor and lastly information leakage, someone can remotely discover information that you don't necessarily want them to have.

And all of those things we see on a regular basis. The statistics out there, our security research team right now, we have about a 93% penetration rate on applications, so what that means that is that obviously something's not being done right. There are a lot of vulnerabilities that are out there that are waiting to be exploited.

So what does that mean for the business? I have to say that the driver for most of us is organizations want to keep their name out of the front page of the newspaper, or out of the television news or out of the mainstream media. They are most concerned about brand and unfortunately that's something that's difficult to quantify but certainly something that an organization faces. I don't want to name any organizations but if you think just in the last six months, actually in the last few days I could name about three or four different companies that have been compromised because of applications. So typically the brand represents the biggest risk for the organization.

Coming down from that of course litigation is always an issue, liability associated with the theft of data for example, or it could be loss of revenue because of a denial of service attack, all of those things impact the organization.

The second client that we tend to forget about in all of this is the person who is accessing the software and this is the client and really there's no kind of cascading list of most severe less severe when it comes to the client. Typically if there's a vulnerability that affects the client on the other end of the application, any of the issues related to the client become available for the attacker. So for example the first thing is environmental compromise. So if the client is compromised then you might have access to their computer or to their file system or application compromise.

Imagine for a moment that an e-commerce application or it's an application that allows someone to do online banking. If the application itself is compromised then a remote attacker could impersonate the client and do very damaging things. So when a client is attacked there is always the gray issue of, well who's liable for that? Was it the service provider? Was it the person who developed the application? There's all kinds of gray areas associated with that but right now client compromise is one of the greatest risks that exists and unfortunately there's not very much protection for the client.

As you might imagine most clients are running personal firewalls or anti-virus or a number of different technologies on the client side to protect them against certain types of compromise, but when an application is compromised that they're interacting with there is not much available in terms of protection for that client. So that is certainly an area where they are lacking in protections and it's up to the organization to provide them with that secure experience. So all of these things are certainly potential outcomes of vulnerabilities and something that we want to address when it comes to security is part of the development process.

So Brian maybe you can talk a little bit about the Rational Unified Process or processes around developing software and then I can talk about how security might plug into that.

Brian Bryson: Yea sure. So, the Rational Unified Process, okay. So the unified process is what we call an iterative software development process framework, which is really a lot of words for something that's really quite simple. It is effectively a methodology for software for delivery. There is I guess, you know, it's got a rich history and a very long history and it's, you were speaking about brands, you know, the Rational Unified Process, somebody who has a lot of brand recognition, so some people, you know, are familiar with the RUP of years ago and it's really evolved over the years.

RUP used to be our one size fits all methodology based on best practices for software delivery and it was, you know, it was something that came out of our consulting organization based on 20 years of developing systems from, you know, as small as a Web commerce site to as complicated as, there's probably a million examples, the you know, one of the most complex ones was the Canadian air traffic control system. Every airplane that flies over Canada is controlled by software that was developed by following the Rational Unified Process methodology.

So, a lot of people know it from that and, you know, it grew from a day where process was, there wasn't a lot of thought on process, there wasn't a lot options for different processes. Now in more modern times it's really evolved, you know, we now have the (Agile) development; we've got interim development, software development. So what RUP really is today is a bunch of methodology bits, if you will, that are combined to form a process that is adaptable to your organization. So we have an (Agile) version of RUP, a RUP for SOA development, RUP for systems development. We have a traditional

RUP, we have lightweight RUP, we actually have a middleweight RUP. There’s over a 100 variations of RUPs these days that have all evolved as specializations of this one core methodology.

Now all of these flavors of RUP have a couple things in common. There’s the, you know, there’s the principles of them all but I might as well run through them, it’s the ABCs of RUP, A is adapt the process, B is balancing stakeholder priorities, C is collaborate across teams, D is demonstrate the value iteratively, E is elevate the level of abstraction and F is focus continuously on quality.

So there’s these six fundamental principles of RUP that are in every variant of the methodology, but really there’s one. And the one core philosophy of RUP that is ubiquitous is the iterative methodology. Instead of the thinking that drove us to build software by doing a whole bunch of design up front, a whole bunch of development a whole bunch of tests and then release, you know, the RUP revolution and, you know, the core tenant of RUP is to develop iteratively, design a little, develop a little, test a little, release. Design a little, develop a little, test a little, release. It’s almost; it’s got a beat to it. Maybe make a song to it; we’ll get Britney Spears on it or something when she has a moment.

So it’s an area of process and within each iteration you attempt to do all of the disciplines, all, you know, develop, design, test, deploy. And one of the things that we’re starting to, that is implied with that is security testing. Security testing needs to come in at every iteration at every stage. You know design for security, test for security, look for those vulnerabilities so that when it comes to release time you have an application that’s secured from risk.

So that's our, you know, behind everything we do is, you know, the basic steps of, you know, it's the RUP methodology which is, you know, prescribes to the iterative development process where security is injected at every stage and you know, the reason we do that is you know, as much as security is new, security vulnerabilities, and maybe Danny will get mad at me for this, but security vulnerabilities are just defects, they're another kind of defect. A bug is a bug is a bug. And it's whether your system doesn't work because it crashes when you click on this button or your system doesn't work because someone can do a cross site scripting injection on, or an injection on a field or can do some type of security attack at some point in your application.

The bottom line is it is still a defect. And you know, it's just another part of the process that we need to manage. It it's, you know we haven't even modified the unified process to specifically call it security at this point because it's still, they're all just defects is really what it amounts to.

So that's kind of the overview of the RUP methodology.

Danny Allan: Thanks Brian. So, I would agree with you. I won't get upset with you at all, in fact it's been something that I've been preaching now for about, well several years, is that a bug is a bug is a bug and security is really nothing new and the people who want to make it new, it's a bit frustrating because what it means is that it removes the ability from the developer or the people building the software to build good software because they might not understand all the nuances of security, but if they look at security issues as nothing more than another bug, it's something that they can fix, it's about validation of input, it's about, you know, very standard things that they've been doing for a very long time.

And so one of the things that I am very adamant about is that security is nothing more than an emergent quality of the software development life cycle, and so this is really just about quality management. Security is just an emergent quality of the software development life cycle. And one of the reasons why RUP and the (agile) process works so well with introducing security testing into it is because the (agile) process in RUP is about iterative and incremental testing, it's about iterative and incremental builds. And there is no better way to catch security issues as an emergent quality than to have automated into the process the testing for or the procedures and the processes to catch these emergent issues that arise during the life cycle.

So let me back up a bit to something I said earlier and that was that there's no such thing as a perfectly secure system. I want to follow-on with that with stating that the goal that we have then, if there's no such thing as a secure system, is that through this process our goal is simply to protect the system as best as we can do within the constraints of the business. So back during the inception phase or as we begin to think about this software, we need to recognize what the constraints are and whether it's even worth it to implement heavy security measures into this process or to ignore them because sometimes it is better to ignore them.

What you want to do is to make the cost for the intruder of breaking into the system exceed the value of what they're going to gain. Now there's about, there is really three different groups of intruders categorized that we look at on a regular basis, the first is the organized crime or the people who are after the money and those are the people who don't really care whether they compromise the organization for data fast for they compromise the client to steal their identities, they simply want to make money.

The second group is the people who are in it for intellectual property and this might be espionage at the corporate level or nation state level. And the third group is the group that I worry least about which is the people who want to do it for fame and egotistical reasons they go out and they promote themselves, hey I hacked into company X. I worry least about them because the risk to the business is far less than the people who want to have ongoing compromise of the organization. Typically those people are very in your face.

So what you want to do is to make the cost for that intruder breaking into the system exceed the value of what is gained. So how do we do that? One of the things that you have to have is a very strong framework up front to address the emergent issues, the quality issues, forget about security, you need a strong framework up front to address the quality issues of the software and the reason why I believe that RUP fits into this so well is that they address that quality management through the process, through the inception, the elaboration, the construction and the transition.

So one of the things that you need, obviously, is collaboration across teams. I run into organizations that say I'm going to address security, I'm going to plug in this automation here. I'm going to plug in this automation here, I'm going to do this here, I'm going to do this here. And they have ignored the very first thing that they need to do, which is to have collaboration against teams. You cannot have isolated islands.

So one of the things that you want is to have the security team having open dialogs on an ongoing basis with the development teams, with the testing teams, having the people sit down if it's only for 15-20 minutes, open the doors of communication between those groups and if you do nothing technological that is one thing that I suggest that you begin with because you need to open those lines of communication. Then you begin to look at the process itself and what you do you need to adapt some kind of structured process, it does not need to be a heavy lightweight thing, and I'm frustrated by people who look at RUP or look at certain process methodologies as being heavy weight, they don't need to be, it's just a framework for a process and it can be adapted to the software that you're building.

So some of the things that you want to address and you mentioned the ABCs of RUP. I have something similar; I call it the vowels of security, the five vowels. And the reason I mention these is because what I believe is necessary is you need those core construction components for the process methodology, the first for A is artifact collection. You need a place to collect artifacts and most of you if you're using Rational products you're using (Clearquest) or you're using some kind of repository right now to collect those pieces of information. So artifact collection is absolutely essential otherwise you can't measure the quality of the software as it's being built.

The second is enablement and this is not just tool enablement, it certainly is tool enablement, you might want to for example, during the build phase you might want to automate the process of doing testing. If you're using build forge for example, you may want as part of that build process to do some kind of security analysis. It only makes sense and then on the back end of that it feeds the quality issues, security issues, any issues that it finds into (Clearquest).

So tools are a part of it but it's more than that. Enablement for the developers means having a repository of components that they can go to or repository of education that they can go to, it might be Web based training. You need to enable the developers not from a security perspective but from a quality management perspective, and one of the things as a company that we've tried to do, Watchfire, is to turn the focus around and not to talk about (sequel) injection and cross scripting and HTTP response splitting and all these technological jargons because you typically get two questions, well what is that and prove it to me and the second thing is, those things change every single day, there's always new attacks, but to turn it around and to report and trend and monitor and perform the Web-based training on the causes not the symptoms of the issues but the causes.

So, most of those things that I just talked about, for example, are input validation and so in that repository you would have certified components that would do things like validate a zip code, validate a phone number, validate an address that would do things like automate the encoding of a particular variable back to the client. That would be all part of the enablement module. You don't want the developers needing to come up with the security wheel, that's already been invented hopefully by the security team. So enablement is a big piece of this.

The third, I, incremental testing – I don't know how many people that I run into say, okay I'm going to make secure software and they come up with a big long list of 250 things they are going to do. To me that is the wrong approach, I've seen it fail time after time after time. What I tend to encourage is that you do incremental testing. Forget about the 10,000 vulnerabilities that exist right now and focus on the two issues that impact your business and then look for two or three issues that have a common cause and focus on those because if you can have the developer start to look at the causes and it's probably going to be with input validation, if you begin incremental security measures into this process with common causes it's much more palatable and it eliminates all kinds of security issues by focusing incrementally on very basic fundamentals of good software delivery.

The fourth vowel, O, ongoing testing – one of the great things that I love about the (agile) process, the RUP process is that it is iterative and there is no better way to do testing than to do it in an ongoing basis. You do not want to use the waterfall approach where the security test gets done once at the end of the cycle. The best thing that you can do for eliminating, for improving the quality of that software is to have ongoing testing through the process, iterative testing through the entire development process.

And then lastly for U, unify the policy – this is not, don't look at it as a security policy and a performance policy and a functional policy. Unify them all into one policy that addresses the quality of the software. And with those five fundamentals then you can begin to introduce or address the quality within the process of delivering software.

Now I can talk specifically about steps that you can take during the inception, elaboration, construction, transition phases, but maybe I'll turn it back over to Brian for any comments that he wants to make about that.

Brian Bryson: Yea it's pretty interesting. I didn't know you were coming out with the vowels; it's interesting how RUP-ish your vowel approach is. You know one of the things that struck me is you know we've been talking about this iterative process and we haven't really touched on the fact that, you know, the process that continues on into deployment, you know with, maybe one of the differences we have with security testing and regular testing, while I think you could argue both sides is that, you know, a lot of the testing for, you know, functional issues, a lot of it stops when the software it's really not, not really true, there's a lot that goes on because, you know, the process continues.

But with security testing with the ever-changing nature of it all it's something that really, I mean correct me if I'm wrong Danny, it's something that has to keep going on right?

Danny Allan: Yes that's absolutely the case and a good example of that about two weeks ago there was a vulnerability, well it wasn't new but it was kind of promoted to the public awareness about issues that arise within flash files and that is something that if you don't have an incremental testing or an ongoing testing procedure after the transition phase, once it's actually in deployment, then you might miss vulnerabilities like that that arise on an ongoing basis.

Brian Bryson: Yea and you know that really speaks to our whole, you know, it goes right back to the beginning of the continuous process powered by automation to govern the software delivery. You know the whole process includes deployment and perpetual checking or, you know, for you ongoing for your O val, you know, for me our focus on continuous quality, it's you know, it's all really saying the same thing.

So, you know, maybe I'll pause at this point. I notice Angelique you haven't interrupted us with any questions so either people are not star 1-ing to ask questions or we've been talking too much and not letting you get anything edgewise, so is there anything in the question queue for us Angelique?

Angelique Matheny: (Robin), can you open up the call for questions or do you have any right now?

Operator: At this time I would like to remind everyone in order to ask a question press star then the number 1 on your telephone keypad. We'll pause for just a moment to compile the Q&A roster.

Again, to ask a question press star 1.

Angelique Matheny: I also have a few questions that came in prior to the actual beginning of the call, I sent out an e-mail so we can get started with those if you'd like?

Brian Bryson: Yea sure, perfect.

Angelique Matheny: Okay. Here's a good one so start with. Here's speaking of RUP, I don't currently follow RUP, can our development shop still adopt some of these practices?

Brian Bryson: Oh, you know it's, you know it's an interesting question so you know the simple answer is yes, you know these tools, you know, fit into a process but you know, don't absolutely require you to follow RUP, but I would almost challenge the statement that I don't follow RUP because there are so many variants of RUP, you know, the definition of what RUP is is that, you know, it is the process you follow, I mean essentially if you've documented an iterative process, you know, whether or not you're actually using our Rational method composer to do that, you're probably following at least the spirit of RUP, so it's very few people today that are still stuck in that waterfall methodology, that's probably the only large category of people that aren't following RUP or at least the spirit of it to an extent.

So I would probably challenge that statement if I'm not currently following RUP and probably question the caller or the e-mailer but they probably are in some extent or another but regardless, yea you know, our functional test, all of our quality management tools have no requirement for process. Now quality comes from, you know the process and how you implement it. I think it's probably a bad strategy to take a tool I know Danny didn't really talk about the tools and one of our primary security testing tools is (Abscan), take it run it once, you know, get the green flag that you're secure and then release. I think, you know, that's a bad idea but it's certainly something you could do but I think if you have a methodology or a process of integrating that iteratively in the continuous automation then yea, you know, you can use these tools to great effect.

Angelique Matheny: Okay great. (Robin), do we have someone?

Operator: We have no audio questions at this time.

Angelique Matheny: Well, then we'll just continue with the ones I have. Danny I think this might go do you. We have a team of very talented software developers but they're not security experts. Can they be expected to use these tools and know how to action their results?

Danny Allan: The answer to that is absolutely yes. Now keep in mind that there is kind of two kinds of issues, quality issues related to security that arise in software. One is kind of very much a functional technical deployment issues, and those are the types of things that automation, as Brian's been speaking to, having continuous automation testing is very good at picking up those issues that arise. And that's because those are mistakes essentially, developers aren't perfect as much as we would like to be, in building software but they do make mistakes.

And so continuous automation to test for those things they don't need to be experts over how to inject a particular script to cause cross site scripting or figure out how to write the correct length of string to do a buffer overflow and then inject a payload, they don't need to do those things because that's what the automated tool does for them. Now the great thing about the tool set is that they don't even report necessarily on the security issue, which arises. Is that important to track and trend? Yes it is for the security auditor, but what they do is they take that issue that they found and they translate it into developer speak or into quality software speak. So no longer is it talking about a buffer overflow, it's talking about validating the boundaries of the input and it talks about it simply from a perspective of having quality software.

There's a second level of security issues, not so much technical as it is by design, and these are issues that fundamentally the product during the inception phase, back when you were doing the requirements and the business model and something made it in there which by design is a very poor design. These are things that continuous automation is not necessarily designed to pick up but these are things that the security audit team should have automatically put into your requisite product, if you're using RequisitePro or something like that, should automatically have those security requirements as part of every single plan to catch these design level issues that arise during inception.

But you'll notice in both of those things, both in the kind of the functional vulnerabilities or the design vulnerabilities versus the technical vulnerabilities, these are things that automation should automatically pick up, the developer does not need to be an expert at all, and in fact the report that he receives has nothing to do with security, it has to do about with building quality software which is their job and which they're very talented at doing.

Angelique Matheny: Thanks. Danny I have another few questions. How are testers supposed to be able to keep up with all the latest hacking strategies?

Danny Allan: So I spend about the first hour and a half of every single day doing nothing but reading everything that occurred overnight, and that's because during the business hours when I'm working I'm reading about vulnerabilities that literally come out on an hourly basis and the research team here, that's what they do, they keep on top of these things.

Let me tell you right now that there is no way for developers to keep on top of the latest threats. And as long as the richness of the environment increases and we certainly are seeing exponential growth with (Ajax) and the Web 2.0 and all kinds of new complex technologies that we didn't even dream of five years ago, they're not going to keep on top of the threats that face those types of applications. Now the good thing is that most of the software packages like Rational (Apscan), that's what the research teams behind those products do. They feed those on a daily basis with all of the new vulnerabilities or the new threats that the applications face.

So the developer does not need to keep on top of those issues, all they need to know about is to, all they need to ensure is that the product is, does have access to those daily updates on a regular basis, and what you'll find is though the threats change on a daily basis, the causes for those threats have not changed in 15 years. It's the same root causes that cause poor quality software today as it was ten years ago. Sure the threats change and those are being dumped into the product every single day and it's being automatically updated, they don't need to worry about that, the root causes don't change.

And so the developer is not going to be surprised one day with a, oh you need to do something completely and radically new because the task that they are given from those updates remains the same, validate input and code your output, very standard things that they should be doing from the very beginning.

Angelique Matheny: You may have just sort of answered this but is security testing something testers can do or something for developers or both?

Danny Allan: Both. And we have seen the most successful organizations are the organizations that actually transfer the responsibility from the security auditor first because they usually start off, they have the security knowledge, and they help in continuous automation with the functional testing team or the QA team or what, unit testing team, whoever it is, they transition the ability to do that testing to the testing team by means of enablement, my E of the five vowels.

And then the more successful organizations, again, continue that migration so that not only do they enable the testing team but they enable the development team. And this is where quality management is going, it is the enablement all the way upstream to the developer themselves to do that testing on their own. So the testing doesn't even necessarily get done in a Rational (Apscan), nor does it get done in a functional tester, but it gets moved all the way up into the developer IDE so that the developer themselves are enabled to produce quality code, and of course the earlier you do it in that life cycle the cheaper it's going to be for the organization because you open less tickets, it's just a much faster turn around time which is what an agile process is all about.

Angelique Matheny: Brian did you want to add something to that or?

Brian Bryson: Well yea, I know we're coming up on our time. It's, it was a very interesting question is, you know, the fact that, you know, the tools makes security testing something that, you know, quite frankly is so fast changing and so really complex in one sense and that, you know, how can you possibly keep up with all this stuff manually. Really is a perfect example of implementing our strategy of continuous process powered by automation, you know, it's continuous in that it's done at the developer level, it's done at the testing level, it's done at the deployment level, it's all throughout little pieces here and there all coming together. You have your continuous process powered by automation because there's just no way that you can manually keep up to date with all of the different security risks that manifest themselves because there's just too many people out there trying to, coming up with these new attacks.

So you know, the implementation of security testing into quality management is a perfect example of a continuous process powered by automation all to govern the process of software delivery, it takes us right back to where we started off this call, I mean we're really trying to deliver applications that are fit for use on time and on budget and you know, we do that with the continuous process powered by automation and this is just one very tangible, very real modern example of how we make that possible and that's, you know, that is how we've evolved from a testing to a quality management organization here at Rational and it's really a much more effective strategy for application delivery. I think, so I'm just thrilled somebody asked that question, it's a perfect way to end it.

Looking at the time I just wanted to get one last thing in, or was there anything on the voice calls? Are we clear there?

Operator: Yes sir, there are no questions at this time.

Brian Bryson: Okay, the last thing that came in to me from one of our internal guys who sent me a pop up message is to talk about just for a second about where we can, you know, where people can go for more information on this (unintelligible) I don't have, I'm not going to list out any URLs or anything like that, but you know the one source stop that I would suggest and maybe can get some data out of this as well is DeveloperWorks is our main stores for (unintelligible) so on DeveloperWorks you'll find information about Webinars and all the rest of it and white papers and technology notes like that for quality management as a whole.

The other, well actually Danny maybe I'll pause there. I'm unclear on the current status of our different Web sites, Danny. Where's the best spot to go for, you know, maybe some of the technical and business information on security testing and maybe even (Apscan) and in general?

Danny Allan: So you can find information on that, I'm going to tell you where you can find security information and then I'm going to tell you not to go there, if that's okay. You can find the Rational (Apscan) product and all kinds of information on the latest threats and security issues and that type of thing at Watchfire.com and we have a zone there that we call the security zone and you can find all kinds of security issues. However I'm going to also say that I think that for this group especially, if you're not a security auditor that it's the wrong thing to do. People are always asking, you know, how do I learn how to be a hacker? How do I learn how to do this and that and the other thing? And because it has some kind of cool factor to it, well, Hollywood has done that unfortunately.

This is not about security. Security is an emergent quality of the software development life cycle and learning about the latest HTTP response splitting or request splitting or whatever it happens to be is not going to help you build better quality software. So though that you might find that fascinating, leave that as your night visits when you're home. If you want to build better software DeveloperWorks in my mind is a much better place to go because this, you know, building good software is just about controlling the future rather than letting the future control you and DeveloperWorks talks about managing that iterative process in a much better way than learning about security is every going to help you.

Brian Bryson: Perfect. The, I'm glad to see that everything has now been centralized onto DeveloperWorks. Another, you know, the other resource and probably the last thing I wanted to bring up here is another great place, a source of information to go is a live event that will be holding in June, June 1st to 5th in Orlando we'll have our annual Rational Software Development conference and as part of that conference we have I believe it's a total of 48 tracks on quality management and I believe security has become such an important focus of that that one-third, 16 of those, or a third of those sessions is all related to the topics of security testing. So it's a, yes (unintelligible) Danny.

Danny Allan: I was just going to say yea that's correct. There is actually two people in the security research team here that are going to be speaking at that event so there is certainly, security is coming to the forefront.

Brian Bryson: It's been, it's certainly going to be front and center. We've got development staff, we've got field staff and technical representatives, we've got demonstration stations where, you know, some of the guys that actually go do the consulting are there to, you know, talk about experiences and strategies and share best practices. We're expecting upwards of 3,000 people to be at this conference to, you know, share information and network. So I think it's possibly the best opportunity that we have to connect and share the latest information on quality management and security topics specifically all at one spot and hey, it's in Orlando at Disney, so it's a nice break from sitting in front of the terminal.

Angelique Matheny: That's right Brian, it's at the Swan and Dolphin, there are over 300 sessions and technical workshops that are available once again.

Brian Bryson: Spectacular.

Angelique Matheny: The URL for that, if anyone's interested, is www.ibm.com/rational/rsdc.

Thank you very much Brian and Danny. This was a very valuable session and we appreciate you sharing your knowledge and experience on today's topic.

If you would like to listen to this conference again or share it with your colleagues this will be made available for replay in an MP3 format in the next few days. To continue the conversation on today's topic we have an upcoming Webcast featuring Brian again and (Morrie Titlebaum) on February 21st entitled Calling All Testers, find application vulnerabilities early in a development process where are easier to fix and less risky to your business. And also for more information or for more Rational Talks view series check out the Rational Talks view page at ibm.com/rational/talks.

This includes the other titles in this series so be sure to join us on February 28th at 1:00 pm Eastern to talk about best practices, integrated life cycle with RPM and Rational portfolio.

Again I would like to thank our speakers Brian Bryson and Danny Allan. Thank you so much for taking the time out of your day to talk to us about testing best practices. I really appreciate you coming.

Man: Our pleasure.

Angelique Matheny: We would also like to thank you our audience for your interest in IBM. We hope to see you back for another one of our events in the near future. Thank you very much. Talk to you soon.

Operator: This concludes today's conference call. You may now disconnect.

END