IBM

Moderator: Angelique Matheny
June 18, 2009
12:00 pm CT

Coordinator: Welcome and thank you for standing by. At this time, all participants are in a listen-only mode. After today's presentation we will conduct a question and answer session. At that time, you may press star then 1 on your touchtone phone in order to ask a question in conference.

Today's conference is being recorded. If you have any objections, please disconnect now. And I'd like to turn the call over to your host, Ms. Angelique Matheny. You may begin.

Angelique Matheny: Thank you very much. Hello, everyone, and welcome to this Rational Talks to You Teleconference Streamline Software Delivery to Gain Market Advantage series.

This is Episode 2 of the series, The Software Blueprint, Designing a Software Solution that Meets Stakeholder Needs. I'm Angelique Matheny. And I'll be your host for today's call.

Now these calls are really for you. We want this to be interactive. And this is your chance to get your questions answered and to discuss what's on your mind.

We'll open up the lines during the Q&A portion of the call. So as the operator mentioned, press star 1 and your line will be open. So write those questions down and get ready.

Also, if you'd like to submit questions to our panelist after this teleconference, please email us at askusnow@us.ibm.com. That's a-s-k-u-s-n-o-w@us.ibm.com. Just put the title of this teleconference in the subject line.

Transforming business requirements derived from stakeholder's demands into a solution that implements these needs is a vital step in our software life cycle. This is where a project's architecture takes the business needs and turns them into the solution.

The solution architecture discipline has progressed by leaps and bounds in the past few years, while agile modeling helping businesses speed up delivery, improve planning and to reduce cost from conception through design and deployment.

And today we'll learn how to transition from work data requirements to an architecture that maps them to the solution.

Scott McCorkle, Senior Manager of the Architecture Design and Construction Product team will talk about the newest developments in architecture and modeling landscape for the systems and IT software round.

And here to discuss data driven architecture is Anson Kokkat, Worldwide Product Manager, InfoSphere Data Architect. Well, with that, I think you've heard about enough from me. So let’s get started.

Welcome Adeel Omer, Marketing Manager of the Architecture Design and Construction Product Team. Adil, why don't you take it away from here?

Adeel Omer: Thank you, Angelique. So as Angelique said, this is actually Part 2 of a series of telecons. The previous telecon was around gaining consensus with stakeholders.

So as all of you are aware, when you start an application or solution delivery process, the first thing you want to do is tie down your requirements.

The next process is to take those requirements and convert them into a solution architecture that maps to those business needs. And that’s where the title for today’s telecon comes in, the Software Blueprint, Designing a Software Solution that Meets Stakeholder Needs.

So I'll go ahead and start off with a couple of questions that we've received over the past week and go ahead and as Scott and Anson a few questions to lay the ground work before we open it up for questions.

So Scott, let me begin with asking you a relatively basic question that connects it to Webcasts or the telecons. What exactly does it mean when we say, you know, we'll create an architecture based on requirements?

Scott McCorkle: Thank, Adil. You know, I guess the easiest way to look at this is to think of the requirements as a set of specifications.

You know, the application must do this. It must not do that. It must do this in a certain way. And, you know, those requirements are usually, you know, they begin their life as sort of a set of simple statements.

You have people sitting around in a think tank saying, "Well, what is it that we want our application to do?" Somewhere you've got to take those simple statements and you have to formalize them, of course, and capture them. But you also have to build them out with a lot more precision.

It's one thing, you know, to say it needs to do this and it needs to that and it can't do, you know, something else. But it's something else to really consider how everything has to fit together and work the way it's supposed to.

During that building out process, they usually will identify several new requirements, you know, that had initially been overlooked or that maybe just, you know, didn't really seem appropriate at the time, you know, based on their original objectives.

So the way that these requirements have to coexist and interrelate with each other and within the environment that they have to function really defines the application architecture.

Obviously you have to specify the way the requirements form the application and you have to really think about their behavior and their functionality for it to be considered a working architecture.

An example for that might be a global accounting system, you know, within a company, where maybe they have a requirement that states the payables are to be tracked in local currency.

And then a separate requirement from a different team states that all purchases must be tracked in U.S. dollars.

If you think about it, this creates several areas of potential conflict that really has to be understood and resolved for that overall accounting system, the overall architecture of that system to function properly.

Once that architecture is really defined and understood, the application software itself can be specified and created to implement the architecture.

Adeel Omer: Okay, so if I hear you right, Scott, that sounds like stuff that's being done for a long time now. Why is there this new focus on and sudden interest in modeling?

Scott McCorkle: Well, yes. And that's a good point because, you know, we're here today to talk about a model driven approach to this and yet nowhere does it say that we have to use modeling. Well, people have been using modeling forever.

You know, visual models have been around for a long time because they're a perfect medium for not only elaborating the requirements and describing the architecture, but just for getting your thoughts down.

You know, it's one thing for one person to have a perfect picture in his or her mind on how it's supposed to work. It's another thing to make sure that that picture matches up with somebody else's same visualization and can be communicated to a whole team of people.

So whether you're using pencil and paper, you know, a whiteboard, you know, or flowcharts as we used to use, you know, these IT applications have

been modeled for a long time so that those ideas could be clearly represented and communicated to others.

But with the advent of service oriented architectures, you know, software as a service, Cloud computing and all of these new IT concepts that are evolving, the applications themselves have become very complex.

And suddenly even trying to draw it on a piece of paper or put it on the whiteboard has become nearly impossible.

Now if you're going to capture that in a model, you know, you're really talking about pages and pages of drawings to really accurately convey how everything is supposed to work.

And it isn't, you know, enough anymore just to say "Draw me a picture or develop a model and, you know, then we can look at it and make sense of it."

Now it's become develop a model and apply some best practices onto how that model has to be built out and analyzed and turned into a working application. So there's been a lot of new modeling innovations over the past few years that have really made today's IT developers stand up and take notice.

The ability to work with the main specific languages and patterns-based engineering, deployment modeling, you know, better traceability back to requirements, better ability to do analysis or impact analysis., all of that is saying that now a formal model driven approach makes a lot more sense for developing these IT applications than it used to.

And the future for modeling, you know, continues to look better and better. When we look at some of the things that are coming down the pipe like model based collaboration over the internet, you know, where teams can actually do their what-if modeling as a group, you know, even though they're not in the same room, they might as well be. And get immediate feedback during their brainstorming session as to what their architectures are supposed to look like.

Adeel Omer: So, Scott, if you're talking about things coming down the pipeline and, you know, things moving to different models, you know, you're the subject matter expert for, you know, from Rational in this space.

So is Rational doing anything to change the way it does business and to meet these needs now?

Scott McCorkle: Oh, definitely. And we have been for a long time. Several years ago Rational pioneered a formal model based approach to architecture specification and that was known as Rational Rose. It was the industry standard for many years. It was a great tool in its time.

You know, it didn't really provide all of the analysis capabilities that teams need today, but it provided the hooks to actually develop those architectural models into a common format, you know, that could be used going forward.

We've evolved on Rose to a product that's called Rational Software Architect or sometimes some people know it as RSA, where we develop real support for IT architects that want to create a working system architecture directly from the requirements and be able to trace their implementation back to the requirements.

You know, RSA uses a set of industry standard diagrams to really help input the requirements into a complete model which fulfills four basic objectives.

And the first is that it translates the requirements into a clear picture of what is supposed to be done, you know, with the focus on the intended functionality and behavior of the project.

Second thing it does is it clarifies the confusion between the requirements themselves and it helps define a thorough list of the requirements that are needed to implement the entire project. Third thing is it attains a mutual understanding of the requirements amongst all the involved stakeholders.

It's one thing to build out a system that meets all the requirements. It's another thing to make sure that everybody understands what those requirements are. It's very easy for requirement in text form to be misinterpreted and misunderstood.

This way they can actually see how the requirements operate, how they work, how they're supposed to work and agree as to how that functionality of the individual requirements behave as well as of all the requirements working together.

And finally, and maybe most importantly, especially if you're in a more regulated industry or more regulated company, is that RSA uses traceability from the final project features back to the originating requirements to prove that each requirement was satisfied and tested and that no additional unnecessary requirements or features were added.

Adeel Omer: Okay. So going back to the title for today's telecon, Scott - and this is my last question to you before I just ask some questions of Anson - you know, you

talked about these capabilities in RSA but how do these actually help us develop from requirements that have already been specified?

Scott McCorkle: Sure. That's a good question. You know, the project architect - and, again, within different groups those people have different titles - but basically they're the ones responsible for putting together that overall application architecture.

They're the ones that can analyze this model, this requirements-based model to verify that the requirements are being satisfied and validate that the architecture functions the way it's supposed to.

So RSA gives them the tools through extensions, you know, and capabilities that are built on top of its model drawing capabilities to really perform that analysis, to simulate functionality, to understand and see that it's really working properly.

You know, then she or he can observe the intended functionality and behavior to determine the next level of more detailed sub requirements. They can say, "Okay, it works at a high level. You know, all the pieces are there."

But obviously the devil is in the details. You know, how do we really make this part of the application work properly?

And so those are the sub-requirements that they start specifying. They feed those new requirements back to their formal requirements management or tracking solution so that they can be incorporated into the specification.

You know, there's all kinds of change analysis and permissions and things that have to be performed before any new requirements can be added to a

specification, you know, for pretty much any company that follows a real process in its development approach.

And finally, you know, with those requirements back in, you know, as part of the specification, they move forward again and start looking at that more detailed model.

And, again, build out the requirements below that so that they iterate in this type of approach, in this type of fashion, until they have a real detailed and complete application architecture where they can look at the high level picture. But they can also go down and study what's happening at any point of the application, in any small subset of it.

And they can analyze whatever changes are made at any subset, you know, see what effect those changes have on the overall application, very important for impact and regression analysis to be able to do that. So let me go back to the example of the accounting system.

The architect, you know, has this what appears to be, conflicting statements on currency handling for the overall accounting system. And the architect can now develop these separate use case scenarios for the systems that show, you know, that meet those individual requirements for currency handling.

You know, he can, you know, look at the impact of these conflicting requirements to see will these two different modules that have two different currency requirements ever communicate with each other or communicate with a common source?

If so, will we have problems with translation? How do we rectify those problems? Do we base it on using the most current currency exchange

standards? Do we do it based on some predefined currency exchange for each type of currency?

How do we address those requirements? Or do we go back to the specifiers and say, "Guys, you have to get your act together. We can't meet your requirements because this just won't work together."

That lets them to really view that overall system impact, look for coexistence, look for functionality, look for ways of resolving those conflicts.

And, of course, they can make their own suggestions as to proposed changes to the requirements, send those back to the specifiers and, you know, with a recommendation as to how they might proceed where it would fit into that system's model and have that considered for incorporation.

So this iterative approach to architecture, you know, to the point of completeness now lets the project, you know, be deemed ready for the next phase which is start defining and developing the actual software.

You know, right now we just have an application architecture. We still haven't written our Java or whatever based software to really implement that architecture. But the good news is the model, in RSA forms the specification for that software.

The relevant portions of the model can actually be pulled out and handed to your software team, who can then, again, iteratively build out those software models to a point of completion to where their software itself is completely specified. And if they want even they can generate real production code from that model.

As they do all of this and as it moves into that software phase, any changes in the project at any level, whether it's in the, you know, change in requirements because the company has new accounting standards that they have to address, or maybe it's because, you know, they've suddenly made a switch in the software compiler that they're actually using for their java code or the application server that it's going to run on -- regardless of where that change is, they can analyze the impact of those potential changes throughout the application architecture, throughout the requirements to see what impact those changes will have.

And make a go, no-go decision long before they have to actually do all the work and find out, you know, months later that it was a disastrous choice in the first place.

Anyhow, yes, that's a little bit long-winded, but RSA does provide that type of capability. We're very excited about it. We have some great customer stories that really talk about how they've achieved the success. And so, Adil, I guess that'll just comprise my answer for...

Adeel Omer: That was - no, that was great, Scott. I mean, I'm looking at this a little differently now to where, you know, architecture turns out to be this stage where you kind of refine your requirements, figure out what's going wrong. Instead of just jumping from requirements straight over to writing code and figuring out months later, you know, hey, we should have made the different decision a long time ago.

You know, it goes back to your point of impact analysis. Now, you know, I'm sure somebody on the call might have some more questions on that. But before we open this line up for questions, of course, we've got the other speaker.

Anson, you're here. You know, we invited you, you know, to be the expert on data architecture.

But just to kind of see where it fits into the big picture, Anson, how exactly does data architecture as, you know, as a term and a practice fit into the software architecture practice that Scott's been talking about as well as, you know, mapping from requirements over to architecture?

Anson Kokkat: Sure, no problem. Thanks, Adil, for having me. So I think before I get into that I just want to describe, kind of dive into what Scott was saying about what a requirement really is, right.

A requirement comes from the real world that it's really trying to address some type of problem. And from that problem is where this reference for this requirement comes from. So there's two roads that you can approach this requirement.

You can approach it from a data-centric point of view or you can approach it from an applications point of view. And this is really going to depend on what background you actually come from.

So if you approach the design of handling these requirements from a development background, you come from a development background, you're typically going to use something like UML to capture this. Right?

Whereas if you come from a data background, and that's how you would begin to approach this problem, then you're going to look at things like ER diagrams or entity relationship diagrams. So at some point you want these two worlds to actually tie in together. Right?

And so that's where the domain model in UML supports - in support of the software architecture and you can take this form and take the UML support, the UML form of this software architecture and transform that into a logical data model in the data architecture world.

So I'll talk a little bit more about what a logical data model and how it fits in a little bit later. But essentially this ties in the two worlds together.

So what this means is this really aligns what you do in the software application space with the data that is used to support the software asset. So how does IBM help with all this?

I mean, in the IBM world we have the Rational RSx product that supports the software architecture. And we have InfoSphere Data Architect that supports the information architecture.

So InfoSphere Architect - InfoSphere Data Architect and Rational data architect are really the same product but it's really IDA and InfoSphere Data Architect is the part that takes care of the whole information architecture perspective of things.

And then to tie these things together, we have various transformations that actually let you take like a logical data model and covert that, for example, from IDA to a UML information model in the RSx products of vice versa.

You can go the other way around. You go from a UML information model to a logical data model in InfoSphere Data Architect.

So depending on which background you come from, you are either going to take the approach of taking a software architecture point of view or a data architecture point of view. But really the things that IBM provides allows these two things to actually talk to each other.

So I want to spend a little bit of time and dive more deeper into the data architecture aspect of this. And say, you know, what approach would you take to approach these requirements from a data perspective?

So the easiest was to define data architecture is probably to explain what an actually data architect does, right. Data architects are responsible for - they're essentially responsible for the quality and consistency of their data.

So they do this by understanding what business users and what the application developers want. And their job is to come up with a model that defines the data around those requirements.

So they want to be able to manage changes well. Right? These requirements are not always set in stone.

There is a lot of change and a lot of deviation that comes from these requirements and we want to be able to take into account those changes to be able to factor those in the requirement stages as well.

So if you look at the various tools and the different ways that a data modeler uses to manage data models, probably the logical data model is the most prevalent.

The logical data model is essentially a place where you can store policies, standard metadata, anything, even semantics for the whole organization.

So you can think of the logical data model as kind of the central place where you can store all this stuff. Right? But there's more to that in the logical data model. You can also define your business terms. You can define the relationships between those business terms.

And really what you're trying to do here is have a way for business users to define these business terms and these relationships really independent of the database.

So you don't have to worry about all the data types and anything, whether this is going to Oracle, whether this is going to DB2, et cetera. It's only interested in the data, what forms that and all the different relationships around that.

Adeel Omer: Okay.

Anson Kokkat: You can also make the logical data model a little bit more - it houses things like the policies and standard.

It can also house terms and meanings, security models, privacy models, capacity and growth metrics, so a whole bunch of things you can keep inside of these logical data models.

So if you look at data architecture as a whole, you can think of data architecture as really one of the pillars that you need as an architecture domain to fulfill this whole enterprise architecture (dune) that you're trying to accomplish.

Adeel Omer: Okay. So if I'm hearing you right, you know, when the data architect is figuring out all these relationships between the in-house data and all, there needs to be some sort of strategy, a data architecture strategy, so to say.

So is that easy to do? Are there problems that most people face when coming up with such a data architecture strategy?

Anson Kokkat: Yes. That's a good question, Adil. So I think to answer that, before I get into the data architecture strategy, let me talk about, you know, what are the problems with data in general. Right?

So if you look at, for example, Accenture recently did a survey where they talked to 157 CIOs worldwide.

And they said that basically 75% of these CIOs believe that they can get a competitive advantage if they are using and managing their data better.

Seventy-eight percent said that they were actively looking for ways to improve how they used and managed their data. But only 15% actually said that they were doing this well, comprehensively and well.

So you can see here that there is problems with how organizations actually handle their data. Right?

And if you look at - if you dig down a little bit deeper into, you know, what are the causes behind those problems, you'll see things like, you know, CIOs will tell you that they have to tailor solutions for customers' existing environments. Whether they're, you know, where it's a whole bunch of diverse databases and they have to tailor to packaged applications.

Another problem they've cited is they have to become more agile in responding to emerging opportunities and stress.

And then the third thing is they're trying to align their IT organization with the business world and they want to be able to collaborate better to help them achieve that.

So let's look at the problems that the CIO sees with the data and let's look specifically now at how data architecture poses its own problem. So organizations often they don't have any documentation or they don't understand the relationships between these different data objects.

So that means that there's a lack of understanding and it repeatedly delays these projects. The organizations need more automation, both for the data and for the relationship discovery test as well as the data design test.

Another common complaint that you'll hear from a data architecture point of view is there is no standardization both from a terminology point of view and for the naming in the organization.

So there's often constant confusion about what the business users want and what they actually get when it's actually delivered.

So driving some type of standardization of business terms goes a long way towards either the goal of managing data quality and bringing back trust in an organization and keeping the IT organization aligned with what the business intends to do.

And I want to bring up one other concern that I've heard is, in a data modeling environment they usually think it’s completely distinct from the other software design and development tools.

So it’s hard to actually share design artifacts and stay in sync. Right? And that’s what you'll see with a lot of the other tools that are out there today.

They’re really meant to be silent and really meant to be a point product that only handles one thing. They don't talk to each other.

So when your data design artifacts are in different formats or there’s different repositories, how do you keep the application design and the data design in sync?

How do you sync up the data and the process design? How do you manage these multiple versions across these different applications, across all these processes and across all these data designs?

So this is where data modeling can help. Right? Data modeling software in particular. And what IBM is trying to do is really bring back the whole vision of modeling and how it can help you in solving a lot of these problems.

Adeel Omer: Okay. So, Anson, you know, one last quick question for a quick comment before we open up the lines for everyone to ask questions.

You know, I heard you talking about data modeling and what companies have to do and the problems that they’re faced with, but, you know, in reality it’s tough economic times.

Everybody’s looking for, you know, what can we do right now to get a quick ROI? We don't want to wait for things that'll bring back money after years. So how does - does data architecture help for that in any way?

Anson Kokkat: Sure. So if you look at - you know, before you can actually start saving money you really need to know, you know, what you have and where do you need to apply certain concepts or certain things that can bring about those savings. Right?

The data architecture actually allows you to discover all your new enterprise solutions and all the content around that and helps you to improve the quality of that information.

So common tools that actually work together, they allow you to foster the IT and the business leaders to define this common model vocabulary and governance requirements that work across the enterprise. So once everyone has this common understanding, you actually start to see real results.

So an example of this is let’s say I'm translating my common vocabulary and my governance requirements into real physical information structures, right, that can be shared across not just the data architects but can be used by the developers, can be used by the testers, can be used by DBAs, by data stewards, a whole bunch of different roles across the organization.

There was a recent report from a leading analyst and it actually says that if you utilize data modeling tools within an organization, you can increase productivity gains by about 20%. So that’s real savings, the real return on investment that you get.

And it’s really the understanding of this data that helps you to realize that value. So understanding via this data modeling approach does actually allow you to save money right away.

Adeel Omer: Excellent, excellent. Thanks, Anson. So, Scott and Anson, thanks for, you know, for those quick questions and answers. At this point, you know, I guess we'll open up the floor to all the attendees. (Brad), are you there? Can you tell us what we need to do next?

Coordinator: Certainly, sir. At this time, if you'd like to press star 1 to ask a question, you may do so. You may also press star 2 to withdraw your question, if you wish.

Once again, to ask a question, please press the star then 1. One moment for our first response, please. Once again, if you'd like to ask a question, please press the star then 1.

Adeel Omer: Okay, while we’re waiting for that, you know, there was a mailbox so people could send in questions ahead of time and I do have a couple here. So, (Brad), are we still waiting or should I go ahead with the couple that I have? I guess I'll go ahead.

Scott, this one’s for you. And the question is there’s been a recent focus on operational modeling. Can you introduce us to the deployment architecture platform and what its benefits are? So, Scott, is there anything that you'd like to talk about there?

Scott McCorkle: Well, yes, it ties back to the same thing I talked about which is, you know, at any stage of operation in any business the objective, of course, in some form or another is to remain solvent by staying in business.

So as you model out your approach, your business approach, to what it is that, you know, how you want to operate, eventually you have to translate that into deployment software.

You know, in other words, what types of systems and services are you going to deploy to enact that business strategy or even what sorts of products are you going to build in parts of it?

The operation model just specifies how it is that you want to operate as a business. It's simply put. How do you want us to implement a set of best practices to conduct our business and how should that feed into the architectures and the applications of products that we deploy to pull that off?

I don't want to make it sound too mystical or magical. We really try to keep it as simple as possible so that, you know, at any level, any person that is involved in that workflow, in they're part of the operation is doing what they're used to do.

You know, they are modeling the business operations. They are creating the applications. They are creating the software.

The idea is to keep the models that they use in a language and a format that's familiar with them and then translate that information into the necessary language and format for the people who follow them in the workflow or have to review their work from above or are monitoring their work from outside.

It's a great way just to let them work in a notation or a series of notations that best describe what they're doing in the way that they're used to working. And then, as I say, handing those work products off to others who can work in their own notations. Does that answer your question?

Adeel Omer: Yes, no, that's exactly it. And, you know, one of the things I think we talk about a lot is one of the things the architecture helps you with is planning ahead.

As you were talking earlier, when you architect based on requirements, you can do that back and forth to save time up front, so you don't waste time later in the cycle when the code's been developed, and now it's not meeting the original requirements because we never got to clarify the requirements and we didn't interpret them when we were writing the code.

So I guess this is just another way where when you've planned for the deployment of the application, it's not like you've finally deployed it or created the application but there's no infrastructure to deploy.

The operation's focus can start getting prepared, you know, on their end for the operations, the deployment of the application ahead of time.

Scott McCorkle: That's right. I mean, simply put, you know, we're trying to minimize surprises.

There's always going to be surprises because you can never account for what your competitors or your customers or your government or anybody else is going to do that changes the way that you work.

But what you can control is that if you have an idea that you want to implement as a company, whether it's an IT system or anything else, you can control that the resulting products that are built out from that. You know, the deployed software in the case of IT systems, exactly matches the original intent that, you know, launched that project. And it amazes me.

I've been in the business for many years and I'm continually amazed at how many times I see companies where they start out with an idea for a system that they want to build and deploy, and somewhere along the way somebody in development or, you know, whomever decides that they have a better way of doing it and they go off on a tangent.

And they write their own vision, you know, implement their own vision of the software so that by the time it really rolls out, it doesn't work the way the business people actually conceived and demanded. We eliminate that through modeling because we have a series of checks and balances.

Any time you write something or develop something or modify something within the model, it immediately rolls back so that you can show that it still correlates with the requirements, you know, with the other aspects of the system. So that there aren't any of those surprises brought on by, you know, somebody who just decides they have a better way of doing things.

Adeel Omer: Got you. You know, that makes perfect sense. Thanks for that answer, Scott. Quick question for you, Anson. Rational Data Architects and now InfoSphere Data Architects, what's the difference? You know, they're very similar sounding. Can you talk about that?

Anson Kokkat: Yes, so I get this question a lot. I think the short answer is that InfoSphere Data Architect and Rational Data Architect are pretty much the same product. What happened is InfoSphere, we rebranded this to InfoSphere Data Architect in December 2008.

And what we're really trying to do is align this a lot with the full InfoSphere Foundation Tools notion.

So if you go to Google and search for InfoSphere Foundation Tools you can find out that we have a whole bunch of tools that are really aligned around understanding of the data and a whole information agenda perspective.

So this doesn't mean in any way that we're moving away from the deep integration with the Rational portfolio. If anything, we're actually getting better with this.

But we're trying to recognize kind of the central roles that this data architect product plays when you are talking about governing information assets.

And so in short it really is the same product but it's just more of a branding thing for recognizing its role in the InfoSphere Foundation Tools.

Adeel Omer: Okay, okay. So it looks like, you know, it's a segment where we're really focusing on data management. And InfoSphere is going to take a lead on that.

Anson Kokkat: That's correct.

Adeel Omer: Okay. I don't have anything else. (Brad), do we have any more questions from the listeners?

Coordinator: We have had no responses so far, sir.

Adeel Omer: Okay, well, everyone, this is your last chance to ask Scott and Anson a question. Actually, you can send a question later to Ask Us Now as well, the email address that Angelique talked about.

But this way you'll get it faster. If there is nothing else, then, Angelique, I'll hand it back to you.

Angelique Matheny: Great. I just want to give that email address again. It's a-s-k-u-s-n-o-w@us.ibm.com. I monitor that daily so we will get your question forwarded very fast.

So thank you so much, Adil, Scott and Anson, for taking time out of your day to be with us. This was a very valuable session.

And we appreciate you sharing your knowledge and experience on this second episode in our Streamline Software Delivery to Gain Market Advantage Series.

For our audience, mark your calendar for Episode 3, Migration from Compuware to Rational Software Testing Tools and RTTS Customer Success Story.

That's going to be held June 25, next week, 1:00 pm Eastern. Be sure and register today. I just wanted to mention a Webcast that you might find interesting, Real World Examples on How Rational Can Help You Accomplish More with Less.

You can find it on Developer Works under Technical Events, then Webcast. So check it out today.

If you'd like to listen to this conference again or hear it with your colleagues, this will be made available for replay in mp3 format in about a week or so on the Rational Talks to You site, www.ibm.com/rational/talks.

Our previous teleconferences are available there as well. 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.

END