

IBM Podcast

[ MUSIC ]

MATHENY: Welcome to this IBM Podcast, What is Quality Management for Systems? I'm Angelique Matheny with IBM. When you're developing smart products and services, delivering the right level of quality is vital to your business, and managing quality is about much more than testing at the end of the development lifecycle.

The true value of today's smart products and services is often derived from the successful combination of diverse technologies in a complex system of systems, where no single organization is responsible for all parts of the system.

So today, IBM expert Jeff Schuster, joins us, and we'll learn about IBM Rational Quality Management solutions. Jeff, welcome to the podcast. Thanks for joining us.

SCHUSTER: Thank you, Angelique.

MATHENY: We've got a lot to cover today, so let's just jump right in. Jeff, let's start with this. How does quality relate to the systems development lifecycle? In the light of recent examples in the press regarding systems quality issues, is there something related to testing that is fundamentally wrong?

SCHUSTER: Well, I think the question in and of itself is very interesting, because you mentioned two things: you talked about quality and you talked about testing. And the one thing that we should understand is that quality is about more than just testing, right?

Testing teams cannot engineer quality into a product. Testing teams validate what's there, and they validate what's there against some predefined requirements that they might have.

So when we think about quality here at IBM, we look at the development lifecycle and we say, well, wait a minute. There's really a quality lifecycle that starts at the concept stage of the development lifecycle. So you really have two cycles going in, embedded and intertwined with each other, so that each step through the systems development lifecycle has a complementary step on the quality lifecycle side.

So, for example, at the very beginning when you're putting out your concept stage and you're trying to define what it is that you're going to build, well, quality begins right there. When you're laying out your requirements, you want to have your quality team involved in that, making sure they understand what the requirements are, making sure they understand what the requirements mean.

And they can also provide input as to whether or not that's a testable requirement. So many projects come in with requirements that really aren't testable. Right? And if you can't test it, how are you going to validate whether there's quality built into the system or quality designed into the system, if you will?

Then when you think about testing, well, that really comes later in the phases, right? Maybe it's after you've implemented some code, you want to run some static analysis tests against the code to determine how well it conforms to best practices, how well it conforms to coding standards. You want to look at quality and determine whether you've exposed any security vulnerabilities into your code, if you will.

So, when you think about testing, most people think of it at the end of the lifecycle; when we think about quality, it needs to be throughout the entire lifecycle. And that gives you input and information that can be used later on to assess whether or not you're on track to deliver on time, whether you're on track to deliver on budget, and whether you're on track to deliver a product that actually meets the end user requirements as initially laid out for you.

So when we think about quality, we think about quality

management and artifacts as linked to all elements of the development process whether they're requirements, whether they're models, whether they're change management, whether they're build records. That's all part of quality, when you think about quality as a lifecycle.

So, for us, if you've got a lifecycle and quality is that, you need some way of managing it. And we've introduced something called Rational Quality Manager into the mix here that provides a single point of reference so that you have a place to store all of your quality artifacts. What are the results of those code scans? What are the results of linking requirements down to test cases? What are the results of executing those test cases? This all comes as part of a collaborative application lifecycle management solution we call Quality Manager.

As part of Quality Manager, you also need to sort of, you know, integrated reporting so that you've got real-time data available to you to assess how far you are in terms of your implementation phases, how far you are in terms of validating the quality of what you're laying out. And this all allows you to make better decisions, more informed decisions, and allows you to keep the project on time, on cycle.

MATHENY: So, Jeff, I think that leads to our next

question. So, what is requirements driven quality, and more importantly, how do we achieve it?

SCHUSTER: Well, when you talk about requirements driven quality, it gets back to what I had mentioned in the opening dialogue here. It's how you define quality. And when we look at quality, it's, quality of a product is defined by how well it meets the end user requirements.

Many people think about quality and they think about testing, making sure the code that you wrote functions properly. Well, that doesn't necessarily mean that you've got a high-quality product, it just means that what you've implemented works.

But the real question about quality is, is what you implemented, does it achieve the requirements that were laid out by the initial end users? And the only way to understand that is the requirements driven quality approach.

Now, how do you achieve that? How do you leverage that?

Well, when you're laying out your requirements very early in the lifecycle, make sure your quality team is part of that.

Lay out the requirement, link it to some design artifact, link it to some code artifact that will be built, but also start linking those requirements to your test artifacts that you're going to build out, right?

What tests do I need to run, when? How many tests do I need to put against this specific requirement? And now, when you get later in the execution phases, you execute those tests, you find ones that fail, you find tests that pass. You compare this against your requirements. You have full traceability back between what the end user objectives were or the upfront requirements that you have laid out, and the actual product you've delivered at the end of the game.

So specifically it means using this traceability to understand what type of quality you have in the system overall. When you also look at requirements driven quality, it's also an excellent way of understanding how much feature creep you have into the system. Right?

Many times code shows up at the end of a development lifecycle and you say, why is this code here? Who put this? What does it accomplish? Oh, well, we just thought it would be a good thing to add. It makes the product better. Well, making the product better is in the eye of the beholder, and if it wasn't part of your original requirements up front, you really have to question why it was added towards the end or throughout the lifecycle at some point in time. So, requirements driven quality really ensure that you're delivering the product that your end users have asked for.

MATHENY: So, Jeff, how do Rational Quality solutions help with risk mitigation?

SCHUSTER: Well, with Rational Quality Manager we advocate and we provide a process for you that allows you to assign risk ratings right from the very beginning when you start identifying requirements.

If you do this early enough, it allows you to create a priority ranking of what requirements you need to implement, what requirements are most critical for a project, and what requirements can be deferred down the road. That way, when you get to the end of the cycle and the end of the development lifecycle, you can make some informed decisions about, well, should we slip the project a couple of weeks? Or should we slip some function out of this project? How important is this requirement really?

And sometimes when you wait until the end of the project to make those decisions, you've got a skewed view as to how important that specific requirement is. But if you did this up front -- when you're defining what you're trying to accomplish -- now you've got a very clear view as to what's most important and what can be deferred until later.

When you look at this, many, many systems are so complex that you can't really test everything. So you want to

understand, what's the most important things to test. When I get to the end of a lifecycle and I only have three or four days left of testing, what tests are most important for me to run?

Well, you should run those tests that are associated with high-risk requirements or high-risk test scenarios that you identified early in the project. So, we help you understand your risk mitigation by enabling you to define what's most important to you up front in the project. Then later in the stages you execute those high-risk items first, and you can keep the lower risk items lower on the plateau, and you execute them as time enables.

Risk management and mitigation is key to effective cost management, and cost management is really what it's all about when you're trying to deliver these solutions into the marketplace.

MATHENY: And how do Rational Quality Solutions support diverse system testing tools and manual testing needs?

SCHUSTER: Yes, thank you. I think what you've pointed out is something very important, and it comes out in the question itself: diverse system testing tools. When we look at this market -- the systems market overall, the embedded market as well -- you look at all these different solutions

out there, all these customized solutions.

And when we work with our customers, we oftentimes find these customers have their own proprietary testing solutions. They've built their own test workbenches in their lab. They've got something proprietary built up for their very intellectual or their IP property that they have. They don't want to expose that to the market as a whole, because it's something unique to their solution.

So when we look at how we can best help them, we were asking them, well, what do you need most? And they say, well, we have these proprietary solutions but we're not linked to the software development lifecycle. So they said, we really need a management layer, and that's really what Rational Quality Manager brings into the mix here.

It's a management layer that allows us to link to the development lifecycle, allows us to link to the upfront requirements that are laid out for full requirements traceability between project definition and project validation in the testing sense. And then we also linked Rational Quality Manager to all these proprietary solutions that are out there for testing their workbenches, testing their internal deployments.

The next part about this is building up a testing ecosystem.

We've recognized there are a lot of proprietary solutions out there, there may also be a bunch of partners out there who have implemented solutions to help them as well. So we wanted to create Rational Quality Manager as an open ecosystem. It provides a management layer, it has integration to proprietary solutions. It has integration to partner solutions that are out there, all helping people bring higher quality solutions to the market.

Now, that doesn't mean that IBM doesn't deliver practitioner testing tools in this space, certainly we do. We have offerings like Test RT, Test Conductor, Rational Robot, different testing solutions for different phases of the development lifecycle, different phases of testing. And we provide these solutions to augment other test scenarios that are out there or other testing solutions that are out there in the marketplace.

By doing this, we're able to leverage this open ecosystem to build up big [partners]. One of the examples would be working with Wind River in their test management solution. Wind River delivers an RTOS solution set out there, and they've got tools designed specifically for that environment. Well, rather than us try to compete and exchange those tools, let's partner with them and enable them to be a part of a bigger lifecycle.

The open interfaces that we provide with Rational Quality Manager allow us to connect to any real third party tooling that are out there. You know, system testing is diverse, so flexibility is key to building up a bridge to all the solutions that may be out there.

The other aspect of this is manual testing. Many test scenarios cannot be automated. Those that can't be automated, we have to have a manual test bed for them. And that's all delivered as core functionality in Rational Quality Manager.

MATHENY: And our last question today. You earlier mentioned that reporting is an important aspect to ensure transparency and effective decision making. How do Rational QM solutions help in these respects?

SCHUSTER: Yes, when you look at what we bring to the table from a reporting perspective, the number one item is information available across the lifecycle. I mentioned about requirements traceability in test cases. I mentioned about requirements and how they might align with some run-time analysis that was done against code that was implemented, requirements to build verification records that are out there.

It's important to have all this data about quality across

the lifecycle so that you've got one place where you can see how you've matured throughout your development process. But a big portion of this is being able to customize those reports or tailor those reports to individual practitioner dashboards or individual project dashboards.

Myself, I may have a different view of the project and quality means something different to me as a coder, as a development individual, than might mean for a project manager. So you want to make sure that the reporting mechanisms we bring into place have the flexibility to enable individuals to customize them to what they need to see.

The key to that is being able to provide key facts on the progress that might be made throughout the project. And we use something here called work items. We've implemented work items within our tooling infrastructure that allow us to share data with individual practitioners on the team.

So I may create a work item, and I may say, hey, John Doe, I need you to implement this, or I need you to check out the validity of this requirement, and I sign it off to someone.

I can then monitor that work item to make sure that they're making progress on them as we mature from a time perspective.

And they may take them a day, it may take them a week, but when it's done, I have my dashboard available on my desktop that shows me, hey, Work Item 17 is done, John Doe did their job. And that's automatically displayed on my particular dashboard. No one else on the team may care about that, but by having this advanced reporting mechanism and this collaboration vehicle, we can communicate without even talking. It's just, information flows across the team itself.

The other part about that is making sure that you've got the right access to the right data for the decisions that you need to make. And for that, we leverage something called Rational Insight to give us business level reporting. Insight is an analytics engine that allows us to see various data and make decisions in the context of the program we're trying to monitor.

We want to see this progress across multiple projects so that we can make system level or architectural level decisions and a business level decision as to how to advance what we want to do. Clear ROI, that's the key. You need to understand exactly what you're looking for, and with advanced and customized reporting, you can tailor those to the data that you believe you need to see to monitor the progress of your project overall.

MATHENY: Jeff, this was very informative. Thank you so much for sharing your time today to discuss the title of this podcast, Quality Management for Systems. We really appreciate it.

SCHUSTER: Thank you.

MATHENY: That was Rational's Jeff Schuster, Software Product Manager, Rational Quality and Requirements. If you are interested in more podcasts like this one, check out the Rational Talks to You Podcast Page at [www.ibm.com/rational/podcasts](http://www.ibm.com/rational/podcasts). We'll include a link to a great Webcast to help get you started. It's called, Three Keys to Exceeding Your Customer's Quality Expectations. This has been an IBM podcast. I'm Angelique Matheny. Thanks for listening. Keep tuning in as Rational Talks to You.

IBM Podcast

[ MUSIC ] [END OF SEGMENT]