IBM Podcast

[MUSIC]

MATHENY: Welcome to this IBM podcast, streamlined software delivery to gain market advantage. This is Episode 4, reduce risk and cost by improving software quality, function and performance. I'm Angelique Matheny with IBM.

Software testing will consume all the time and resources it's given, but how does one streamline the quality management process, reducing risk and cost? In this podcast, Brian Bryson outlines quality management strategies for test management, test data management, functional and performance testing that streamlines software delivery to gain market advantage.

So let me introduce our presenter, Brian Bryson, Solution Marketing Manager. Hi, Brian, welcome to the podcast. Thanks for joining us.

BRYSON: No, thank you. Thank you for the opportunity to come and speak today.

MATHENY: Brian, the title of this series, of course, is streamlined software delivery to gain market advantage. And the focus of this particular installment is on quality. So, what can be done from a quality perspective to streamline software delivery and gain that market advantage?

BRYSON: I've been doing some internal research here at IBM and I have been pulling quite a group, over a hundred of our consultants, sales reps, trainers, there were some product managers, some of our senior management, with effectively that exact question. Actually, before I dive into that, let me take a step back and talk about my methodology which is based on a book by James Surowiecki.

The book's called The Wisdom of Crowds. It came out three or four years ago, I believe. A very interesting read, I'd highly recommend it. It's based on a simple premise that -- and this is a quote from the book -- under the right circumstances, groups are remarkably intelligent and often smarter than the smartest people in them.

You know, sort of put another way, under the right circumstances if you ask enough people the same question you'll get a very good approximation of the right answer. Now, it's a little more complicated than that; it's kind of hard to summarize a book in one word or, you know, in one paragraph or one sentence.

But basically that's the idea. He's done all sorts of research and, you know, there's some great stories from

Jeopardy, to Eli Lilly, to racetrack betting, all about how if you just ask enough people you will sort of circle in on getting the right answer.

So, sort of following that thinking, I asked a bunch of internal people at IBM how to streamline software development and delivery. And you know, that's a great way to sort of circle in on the right answer. And I got a ton of responses, and frankly, they were all over the board.

It was probably...I think I bit off a little more than I could chew. But after a sort of long analysis it all sort of broke down into roughly three categories of things you can do and which turned out to sort be effectively strategies for streamlining software delivery. And those were...these are kind of my terms for ways of grouping them, but, automation, collaboration and process.

MATHENY: So, why don't you drill down those for us a little bit?

BRYSON: So, exactly where I was going. And what I'll do is I'm going to start with process since, you know, relative to the others it's the one with the longest implementation curve here. We're talking changing a process. We're talking to sort of years in terms of implementation as opposed to days or months.

But you know, in exchange for that significant effort, done right, this is where you see the biggest productivity increase in orders of magnitude. You know, being at IBM, we have access to one of the largest software test and software delivery consultancy firms in the world with lots of interesting metrics.

And from their research, you know, the right process for the right project can increase productivity two times, five times, even 10 times if you're going from worst case the best case scenario. But given that this is a podcast and we really have a short time here, minutes here, a process discussion is really out of scope.

So, I wanted to bring that up more to just sort of allay any fears that process optimization was an oversight. It's not. It's probably the most significant thing you can do, but it's more than we can handle in this forum.

MATHENY: Okay, fair enough. So, what about automation and collaboration?

BRYSON: Right. So now we're talking about things with much faster and simpler implementation timeframes. So, things that we can...are sort of are actionable when you're talking about a little discussion we're having here.

So, if we start with automation, we're talking here about sort of implementation timeframes in terms of days to weeks to get you going with automation. So if you take a tool like IBM Rational Functional Tester, as an example, you can download and install it in a day and be creating good solid tests within a week.

MATHENY:	Brian, for those that aren't familiar with the tool, can you tell us more about Rational Functional Tester?

BRYSON:	Right, of course. So, Rational Functional Tester effectively allows you to automate your tests. So, instead of, you know, you, Angelique, going and manually clicking on the buttons of your application or service, Functional Tester will do it for you.

Now, the way it works is you actually record yourself performing the test actions and it captures those actions into a script form. And once you've got it into a script form, you can edit it, because the scripting language, it's actually just a program, it's Java or Visual Basic .NET, it's up to you to choose, but you have those choices. So, it effectively creates a program for your that does the test and then you can edit it and run it.

Now, of course, the benefit of once it's sort of captured in script or program form is you can run it overnight so you could effectively be testing 24 hours a day. Or you can run it on a second computer beside you or within a virtual image running on your machine. So you're effectively increasing your productivity here.

And while these tests run, you can be running more manual tests or you can be creating more tests. So, effectively what I'm saying is you're using automation to accelerate your process. Or more in line with our title here, you're really streamlining your software delivery.

Now, before I go on, for those new to this type of technology -- and again, trying to stay with the theme of streamlining software delivery -- it's important to use the tool the right way. And we are crossing a little bit over into process here, but in terms of an approach that I guess many have used and found to be successful most of the time, you know, there's no surefire 100 percent silver bullet here...

But a popular approach, or a great way to start, is to automate key tests or what some people call smoke tests. So, they're not the most complicated tests, but they're tests that you have to run time and time and again. Software delivery teams are giving quality assurance teams builds in these days every day, sometimes multiple times a day.

And to actually have to manually run through these basic tests on every build is really time consuming. So, you can automate them and run them on every build and you make it the purpose of these tests to validate that what used to work still does work.

You're not really focusing on, you know, what's been new and added to the code, but you're just making sure that what's been new and added to the code hasn't broken what used to be there.

And the idea being this frees up your testers to manually validate new application or service functionality, or, once the new functionality becomes available that can automate it and ultimately your regression suite grows and grows and grows and covers more of your application.

MATHENY: And what about collaboration?

BRYSON: Right. So, the last bucket of suggestions or group of tips that came in from these consultants, and product managers, and we call them tech reps, the people that go and actually do installations and get people going with the tools, all sort of fell into the bucket of collaboration.

We're talking about things make a team more efficient as opposed to an individual. So, when you think about Rational Functional Tester, [who] we were just talking about, that makes an individual more productive. It frees one person up from running tests to effectively let them do more testing or create more automation while the tests actually run on a computer beside him.

But, here we're talking about, with collaboration stuff, we're talking about things that impact the team. So, accordingly, the more people you bring into the equation, the longer...a couple things happen here, the longer you're looking at for implementation. So, we're talking about strategies here that you can do in weeks or months as opposed to days with the automation stuff.

And then of course as opposed to sort of years, if we're talking about the process. So, we're sort of in between in terms of implementation time and difficulty. But since we're impacting more people, we're also seeing a greater payoff. You know, these collaboration tips benefit more teams. There are things that sort of make the team work better together. And so the payoff is broader, impacts more people and you get a bigger pay off for your investment here.

So, you know, some of the things that came out of here we're talking about the big headline, if you will, we're talking about unifying requirements into test, or requirements based testing, as some people call it, where requirements drive the tests.

And we're talking about asset tracking -- so, keeping track of work items or just things on the project that need to be done or defects. And finding a way to track all of that stuff gives you the benefit of being more organized and lets you know what you can expect from others on your team and what others can expect for you.

And when you have a system where your test plan is always reflecting the most current requirements, is aware of any blocking defects or is accurately and quickly funneling those defects to development for fixes, is aware of new builds when those defects get fixed so that you can run tests...

You now have a lot of...first of all, you're organized and you have control, but that gives you a lot of data for measurements that you can use to make, you know, project decisions.

And there's not a single quality assurance team that doesn't have a meeting once a week. You know, usually Mondays or

Fridays, from my experience, where the project leader or the project manager, whenever the person running the project comes in and says, how are we doing? What's our status?

And you know, there's no single one answer to that question, but when you have a team all working together, all collaborating, sharing data and aware of what each other's doing, then, you know, you have the metrics available to answer that question.

Are we ready to release? How is it looking? What's our defect arrival rate? Are we fixing things faster than they're coming in? So, we got a lot of responses in that category of sort of team collaboration.

MATHENY: You know Brian, that sounds a lot like what's available in Rational Quality Manager.

BRYSON: Right. And you know, there's good crossover there. Rational Quality Manager was effectively built to unify quality teams with customers sort of from a requirements perspective and designers for that matter, business analysts, with developers in terms of defect integration and tracking work items.

And so the tool was built with the idea or the concept of integrating or linking those people and their data. So, if you look at the tool, or if you go watch the great five-minute demonstration that's on ibm.com, you know, you can see how that actually plays out in terms of sort of software support.

You'll see when you launch Rational Quality Manager how you're presented with a dashboard that is continually updated with a live information. So, what builds have been checked in? What defects have been found? What new requirements have been answered or what requirements have changed since you last went to that page?

You have individual to-do lists and team to-do lists, so you can see what people are expecting from you and what you can expect from others. And you know, there's a live feeds off of RSS feeds, so that you can keep effectively everyone in communications and in sync.

In fact, we can even link in Twitter feed to your desktop dashboard so that the whole team is constantly aware -- they have to be using Twitter or other means of communication -- of what's going on and everyone's in communications and everyone's in sync.

MATHENY: Brian, that's about all we have time for, so do you have any final words?

BRYSON: Yes. So, we've already gone longer than I intended so I won't add anything new. But if we go back to our goal of streamlining software, we've covered two strategies that can really help you achieve that goal. There's individual automation tools like Rational Functional Tester which lets you do your functional testing.

In fact, we didn't even talk about Performance Tester for performance testing, or Service Tester for unit service testing. But these tools have short implementation times, and it can really accelerate individual productivity and really help you streamline software that way.

The other was sort of the collaboration strategies for unifying teams. And we've sort of been talking about Quality Manager here, but these things have longer implementation times but sort of a broader impact in terms of greater returns.

You know, the strategies here sort of unifying requirements to test to development is really sort of the overarching theme, and you do that by in one collected spot tracking your requirements, tracking your defects, tracking your work items, everything that's important about a project.

And then, of course, we sort of started the discussion and we should probably end it with process changes. You know, probably the longest implementation times but huge payback potential. You know, we didn't talk much about that just because it's only a podcast.

But people want to dig deeper into that. Probably the easiest thing is just to Google Rational MCIF, which is our Measured Capability Improvement Framework, MCIF. And it's kind of the entry into how we recommend going about a large-scale process change.

So, you can head to our blog at rationaltester.com where we'll be talking about that and some of the other strategies that we've been talking about here. In the next coming weeks, we've got some posts scheduled to sort of address those things. That's our thoughts on streamlining software delivery.

MATHENY: Brian, thanks so much as always for sharing your time today to discuss our fourth episode in the streamline software delivery to gain market advantage series, reduce risk and cost by improving software quality, function and performance. I think you nailed it, we really appreciate it.

BRYSON: Thank you, always a pleasure.

MATHENY: That was Rational's Brian Bryson, Solution

Marketing Manager. Stay tuned for the next in the series, Episode 5, Pack it, ship it, consistent and automated software assembly processes. 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.

We'll include the link for the IBM Rational Quality Manager demonstration that Brian mentioned, so check it out today. 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]