SOFTWARE DELIVERY IS A CRITICAL BUSINESS SYSTEM

Overview: As Rick Slade discusses in his podcast series, organizations should approach their software delivery processes just as they would any other critical business system. An efficient, effective software delivery system is a crucial part of not just keeping pace with your competitors but becoming a leader in your marketplace.

We recently wrapped up the Building a Better Software Development Platform podcast series. Through 15 episodes, Compuware Executive DevOps Solution Architect Rick Slade made the case for treating your software delivery as a critical business system and explained the steps needed to make it as efficient and productive as possible by incorporating Agile and DevOps methodologies. I recently talked to Rick about how he thinks the series will help listeners, why the subject is important, and the experience of creating a podcast based on his years of experience.

MD: What should be the biggest takeaway for listeners of this series?

Rick: For me, the big takeaway is changing the mindset of how you manage software delivery within your organization. It's not a set of disjointed tools and processes. If done properly, and if operating effectively, it is a system.

> *The software delivery system is a symphony of tools working together to produce a desired output.*

We need to start to look at the software delivery ecosystem as a critical business system, and it needs to be managed accordingly. And that means looking at things differently. If you've got disjointed tools and disjointed processes, you're going to have multiple ways of doing things and it's going to be difficult to manage effectiveness and quality of output.

If you've got an application that is critical to your business and you manage it as such, you're going to be more efficient and more effective. "Efficient" meaning that we maximize the output of our efforts, and "effective" meaning producing great output. Both are important in a modern organization's desire to deliver software. We've got to be both of those things in order to meet marketplace demands and in order to compete with our competitors.

MD: What would you tell someone who is on the fence about whether to make the investments needed to build a new software delivery system?

Rick: I think you have to look at it from a financial standpoint. You have to understand, you have to be honest with yourself with regard to where you are in comparison to your peers in the industry. And are you losing market share? Are you being as competitive as you could be? Are you not leading the market because of your inability to get software solutions to the people who are going to use them?

An honest self-assessment of where you are should dictate a desire to change. Change is the only way that you're going to accomplish those goals. And it starts with self-awareness and a desire to understand how effective or ineffective you are. And the best way to do that is through metrics and measurement. Use that information to self-assess, look at how you're doing, as compared to others. With that information a decision whether to invest becomes much easier.

MD: So, using something like zAdviser is really a kind of first step in the process?

Rick: Yes. It will help you establish a benchmark. Creating a benchmark is a great first step and zAdviser can help you do that faster and, I think, more effectively.

MD: When did the idea of DevOps on the mainframe start gaining some traction?

Rick: I came to IBM in 2007 and it was being talked about. DevOps should not be the goal of an investment strategy. The goal is to deliver better software faster. "DevOps" is just a label assigned to a framework, or a culture, to accomplish that. The desire to deliver software better… I started in the '70s, and we were talking about application design, we were talking about tooling when I was 22 years old, out of college. So, it's been talked about forever and will continue to be.

MD: So, it will keep evolving.

Rick: It will. It's been evolving for the 40 years I've been involved in the business, and it's constantly changing. Why are we to think that it won't continue to do so? And if it is going to continue to change, and you do believe that it is mission-critical to your organization—and who doesn't think software delivery is mission-critical these days—then why should you not have dedicated people and start to manage it as a critical system within your organization?

It's a critical value stream within your organizartion. A value stream is those critical operations that an organization executes in order to better service its customers, or its employees, or its constituents. The software delivery efforts within an organization, I think, are a critical value stream and we should be managing it the same way that we manage our other critical value streams.

MD: We're not talking about putting a new system in place and leaving it for 30 years, we're talking about something that will grow, or change.

Rick: That's exactly right. It needs to be a system and it needs to be built on an architecture that supports evolution. That's why I'm such a believer in the open software delivery model. I want a system where I can mix and match parts, but I can also exchange parts for others that are evolved and even better. Because things are going to get better. Compuware doesn't stand still. IBM doesn't stand still. We're always providing new capabilities and features. And so, you want a software delivery system that can accept or adopt those changes as quickly as possible so to positively impact your own software delivery value stream within your organization.

MD: Does the series need to be listened to as a whole in order to be effective? Can individual episodes help listeners who may be at different stages in the modernization of their software delivery systems?

Rick: I think they can. But I think it's most effective if you look at it as a book. I was talking to Sam Knutson the other day and he said, "I love what you guys did with the podcast. It's like an audiobook, and it tells a great story." So, it's quite effective if you consume the whole thing, but I certainly believe that there are topics you can extract, like testing and automation, that are applicable and usable without going through the entire series.

MD: What did you learn in doing the podcast?

Rick: I learned some things about myself. I learned that I don't know as much as I thought I knew, but also that I know more than I thought I did. I was able to cover a lot of topics with my own knowledge, but as I dug deeper into those topics, I found that there's still a lot to be learned.

I think another lesson learned is it's best if you've got passion for what you're talking about. You've got to care about your subject. You can't be successful without being passionate.

To listen to any of the chapters of the Building a Better Software Delivery Platform series and watch the corresponding Q&A sessions, visit www.compuware.com/goodcoding.</em